K.R. MANGALAM UNIVERSITY

THE COMPLETE WORLD OF EDUCATION

SCHOOL OF ENGINEERING AND TECHNOLOGY

Programme Handbook

(Programme Study and Evaluation Scheme)

Bachelor in Computer Applications (BCA) With Specialization in AI & Data Science [Honors/Honors with Research] Programme Code: 201

UNDERGRADUATE PROGRAMME

As per National Education Policy 2020

(Multiple Entry and Exit in Academic Programmes)

(With effect from 2024-25 session)

Approved in the 34th Meeting of Academic Council

Held on 29 June 2024

Table of Contents

Contents

Preface 1
NEP-2020: Important features integrated in the curriculum 3
Categories of Courses 4
University Vision and Mission 6
About The School 7
School Vision and Mission 8
About the Programme 9
Programme Educational Objectives (PEO) 10
Programme Outcomes (PO) 10
Programme Specific Outcomes (PSO) 11
Career Avenues 11
Duration 13
Eligibility Criteria 13
Student's Structured Learning Experience from Entry to Exit in the Programme 15
Evaluation of Learning: 22
Program Structure & Evaluation Scheme 25
Syllabus 36

Preface

Welcome to the School of Engineering and Technology at K. R. Mangalam University. It is with great enthusiasm that we introduce you to an institution dedicated to nurturing future leaders in engineering and technology.

Established in 2013, our School has rapidly evolved into a premier center for innovation, quality education, and skill development. With a focus on imparting advanced knowledge and fostering creativity, we are committed to providing a transformative educational experience. Our state-of-the-art infrastructure, cutting-edge laboratories, and a distinguished team of faculty members collectively create an environment where academic and professional excellence thrives.

Our diverse programs encompass undergraduate degrees (B.Tech, BCA, B.Sc), postgraduate studies (M.Tech, MCA), and doctoral research across all engineering disciplines. Notably, we offer specialized B.Tech programs in areas such as Artificial Intelligence & Machine Learning, Data Science, Cyber Security, Full Stack Development, and UI/UX Development. These programs are designed to equip students with both technical proficiency and a deep understanding of emerging technologies.

At the heart of our mission is a commitment to a curriculum that integrates the best practices from leading global institutions while also incorporating insights from the Open-Source Society University. This curriculum emphasizes problem-solving, interdisciplinary learning, and innovative teaching methodologies, all aligned with the National Education Policy (NEP) 2020.

Our emphasis on industry integration is reflected in our collaborations with renowned organizations such as IBM, Samatrix, Xebia, E.C Council, and ImaginXP. These partnerships ensure that our students gain practical experience and insights

that are directly applicable to industry demands. Elective options across diverse domains, including AI, Cloud Computing, Cyber Security, and Full Stack Development, offer students the flexibility to tailor their educational experience to their career aspirations.

We are also dedicated to fostering a culture of innovation and entrepreneurship through our Entrepreneurship and Incubation Center and initiatives like 'MindBenders,' 'Hack-KRMU,' and participation in the 'Smart India Hackathon.' These programs are designed to inspire and prepare students to become forward-thinking leaders in the technology sector.

Our modern computing facilities and comprehensive infrastructure support advanced research, simulations, and hands-on projects, ensuring that our students are well-prepared for the challenges of the professional world. K. R. Mangalam University is recognized for its commitment to providing quality education, and our alumni have made notable contributions across various sectors, from multinational corporations to public sector enterprises.

We are excited to accompany you on this journey and look forward to supporting your academic and professional growth. Welcome to a community where excellence and innovation are at the core of everything we do.

School of Engineering & Technology
K.R Mangalam University

NEP-2020: Important features integrated in the curriculum

K.R. Mangalam University has adopted the National Education Policy NEP-2020 to establish a holistic and multidisciplinary undergraduate education environment, aiming to equip our students for the demands of the 21st century. Following the guidelines of NEP-2020 regarding curriculum structure and duration of the undergraduate programme, we now offer a Four-Year Undergraduate Programme with multiple entry and exit points, along with re-entry options, and relevant certifications.

- **UG Certificate:** After completing 1 year (2 semesters with the required number of credits) of study, and an additional vocational course/internship of 4 credits during the summer vacation of the first year.
- **UG Diploma:** After completing 2 years (4 semesters with the required number of credits) of study, and an additional vocational course/internship of 4 credits during the summer vacation of the second year.
- **Bachelor's Degree:** After completing 3-year (6 semesters with the required number of credits) programme of study.
- **Bachelor's Degree (Honours):** 4-year Bachelor's Degree (Honours) with the required number of credits after eight semester's programme of study.
- Students who secure 75% marks and above in the first six semesters and wish to undertake research at the undergraduate level can choose a research stream in the fourth year. Upon completing a research

project in their major area(s) of study in the 4th year, a student will be awarded **Bachelor's Degree (Honours with Research)**.

Advantage of pursuing 4-year Bachelor's degree programme with Honours/Honours with Research is that the Master's degree will be of one year duration. Also, a 4-year degree programme will facilitate admission to foreign universities.

S. No.	Broad Categories of Courses	Minimum Credit Requirement for Four Year UG Programme
1	Major (Core)	80
2	Minor	32
3	Multidisciplinary	09
4	Ability Enhancement Course (AEC)	08
5	Skill Enhancement Course (SEC)	09
6	Value-Added Course (VAC)	06-08
7	Summer Internship	02-04
8	Research Project/Dissertation	12
9	Total	160

Categories of Courses

Major: The major would provide the opportunity for a student to pursue in-depth study of a particular subject or discipline.

Industry Driven Courses (IDC): The purpose of our industry-driven courses is to align academic learning with industry needs. Through engagement with industry experts, students receive hands-on training and real-world experience throughout the semester, ensuring they develop the practical skills needed to become industry-ready upon graduation.

Multidisciplinary (Open Elective): These courses are intended to broaden the intellectual experience and form part of liberal arts and science education. These introductory-level courses may be related to any of the broad disciplines given below:

- Natural and Physical Sciences
- Mathematics, Statistics, and Computer Applications
- Library, Information, and Media Sciences
- Commerce and Management
- Humanities and Social Sciences

A diverse array of Open Elective Courses, distributed across different semesters and aligned with the aforementioned categories, is offered to the students. These courses enable students to expand their perspectives and gain a holistic understanding of various disciplines. Students can choose courses based on their areas of interest.

Ability Enhancement Course (AEC): Students are required to achieve competency in a Modern Indian Language (MIL) and in the English language with special emphasis on language and communication skills. The courses aim at enabling the students to acquire and demonstrate the core linguistic skills, including critical reading and expository and academic writing skills, that help students articulate their arguments and present their thinking clearly and coherently and recognize the importance of language as a mediator of knowledge and identity.

Skills Enhancement Courses (SEC): These courses are aimed at imparting practical skills, hands-on training, soft skills, etc., to enhance the employability of students.

Value-Added Course (VAC): The Value-Added Courses (VAC) are aimed at inculcating Humanistic, Ethical, Constitutional and Universal human values of truth, righteous conduct, peace, love, non-violence, scientific and technological advancements, global citizenship values and life-skills falling under below given categories:

- Understanding India
- Environmental Science/Education
- Digital and Technological Solutions
- Health & Wellness, Yoga education, Sports, and Fitness

Discipline Specific Electives (DSE): The purpose of offering discipline-specific electives is to provide students with the flexibility to specialize in emerging and high-demand domains such as Full Stack Development, Cloud Computing, AI & ML, and Cyber Security. These electives are designed to equip students with advanced knowledge and skills in their chosen fields, ensuring they are well-prepared for specialized roles and industry demands in these cutting-edge areas.

Industry project/Research Project: Students choosing a 4-Year Bachelor's degree are required to take up Industry/research projects. The purpose of our full-time, 6-month industry project for final-year students is to provide them with practical exposure by working on real-world industry projects.

University Vision and Mission

Vision

K.R. Mangalam University aspires to become an internationally recognized institution of higher learning through excellence in inter-disciplinary education, research, and innovation, preparing socially responsible life-long learners contributing to nation building.

Mission

- Foster employability and entrepreneurship through futuristic curriculum and progressive pedagogy with cutting-edge technology
- Instill notion of lifelong learning through stimulating research, Outcomes-based education, and innovative thinking
- Integrate global needs and expectations through collaborative programs with premier universities, research centers, industries, and professional bodies.
- Enhance leadership qualities among the youth having understanding of ethical values and environmental realities

About The School

Since its establishment in 2013, the School of Engineering and Technology at K.R. Mangalam University has rapidly developed into a hub of innovation, quality education, and skill development. Our focus is on delivering a transformative educational experience that equips students with advanced technical knowledge while fostering creativity and critical thinking. With state-of-the-art infrastructure, modern laboratories, and a distinguished faculty, we provide an environment that nurtures both academic and professional excellence.

Our school offers a comprehensive range of programs, including undergraduate (B.Tech, BCA, B.Sc), postgraduate (M.Tech, MCA), and doctoral studies across key engineering disciplines. We are proud to offer specialized B.Tech programs in high-demand fields such as Artificial Intelligence & Machine Learning, Data Science, Cyber Security, Full Stack Development, and UI/UX Development. These programs are designed to meet the evolving needs of the industry, ensuring that students are equipped with the skills and knowledge required to succeed in the modern workforce.

Our curriculum is grounded in best practices from leading global institutions and incorporates insights from the Open-Source Society University. It emphasizes interdisciplinary learning, problem-solving, and innovative teaching methodologies. This approach not only enhances students' technical competencies but also

develops their ability to think critically and work collaboratively across diverse domains.

Industry integration is a key component of our educational model. We collaborate with renowned organizations such as IBM, Samatrix, Xebia, EC Council, and ImaginXP to provide students with practical, real-world experience through internships, projects, and workshops. These partnerships ensure that our students are well-prepared to meet industry demands. Additionally, we offer elective courses in areas such as AI, Cloud Computing, Cyber Security, and Full Stack Development, allowing students to tailor their learning experience to align with their career goals.
We are also committed to fostering innovation and entrepreneurship. Our **Entrepreneurship and Incubation Center** and initiatives like 'MindBenders,' 'Hack-KRMU,' and participation in the **Smart India Hackathon** inspire students to develop forward-thinking solutions and entrepreneurial ventures.

With cutting-edge computing facilities, advanced research opportunities, and a focus on practical application, the School of Engineering and Technology ensures that its graduates are well-prepared to excel in their careers. Our alumni have made significant contributions across various sectors, reflecting the high standards of education they receive.

School Vision and Mission

Vision

To excel in scientific and technical education through integrated teaching, research, and innovation.

Mission

- **Creating** a unique and innovative learning experience to enhance quality in the domain of Engineering & Technology.
- **Promoting** Curricular, co-curricular and extracurricular activities that support overall personality development and lifelong learning, emphasizing character building and ethical behavior.

- **Focusing** on employability through research, innovation and entrepreneurial mindset development.
- **Enhancing** collaborations with National and International organizations and institutions to develop cross-cultural understanding to adapt and thrive in the 21st century.

About the Programme

Definitions

- **Programme Outcomes (POs)**

  Programme Outcomes are statements that describe what the students are expected to know and would be able to do upon the graduation. These relate to the skills, knowledge, and behaviour that students acquire through the programme.

- **Programme Specific Outcomes (PSOs)**

  Programme Specific Outcomes define what the students should be able to do at the time of graduation and they are programme specific. There are two to four PSOs for a programme.

- **Programme Educational Objectives (PEOs)**

  Programme Educational Objectives of a degree programme are the statements that describe the expected achievements of graduates in their career, and what the graduates are expected to perform and achieve during the first few years after graduation.

- **Credit**

  Credit refers to a unit by which the course work is measured. It determines the number of hours of instructions required per week. One credit is equivalent to 14-15 periods for theory, or 28-30 periods for workshop/labs and tutorials

Programme Educational Objectives (PEO)

PEO1: Successful professionals in industry, government, academia, research, entrepreneurial pursuits and consulting firms.

PEO2: Able to apply their knowledge of computer science & engineering principles to solve societal problems by exhibiting a strong foundation in both theoretical and practical aspects of the field.

PEO3: Dedicated to upholding professional ethics and social responsibilities, with a strong commitment to advancing sustainability goals.

PEO4: Demonstrating strong leadership skills and a proven ability to collaborate effectively in diverse, multidisciplinary teams to successfully achieve project objectives.

Programme Outcomes (PO)

Engineering Graduates will be able to:

PO1. Core Competencies: Graduates will possess a strong foundation in computer science principles, critical problem analysis, and solution design, equipped with skills for conducting thorough investigations to solve complex challenges.

PO2. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern IT tools including prediction and modeling to complex computer science activities with an understanding of the limitations.

PO3. Societal and Environmental Responsibility

Apply contextual knowledge to evaluate societal, health, safety, legal, and cultural issues, while understanding the impact of engineering solutions on the environment and advocating for sustainable development.

PO4. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the computer science practice.

PO5. **Effective Communication and Team Collaboration**

Excel in both individual and team roles within diverse and multidisciplinary settings, while communicating complex computer science concepts clearly through effective

reports, presentations, and interactions.

PO6. Project management

Apply engineering and management principles to lead and manage projects effectively in computer science contexts.

PO7. Life-long learning: Embrace and actively pursue continuous learning to stay current with technological advancements and evolving practices in computer science.

Programme Specific Outcomes (PSO)

PSO1: Understanding the core concepts, theories, tools, techniques, and methodologies of Artificial Intelligence and Data Science.

PSO2: Applying AI and Data Science principles to solve real-world problems and make data-driven decisions.

PSO3: Analysing data and AI methodologies to uncover insights and address challenges.

PSO4: Evaluating and optimizing AI models and data solutions for enhanced performance.

PSO5: Designing and developing innovative AI and Data Science solutions to tackle complex problems.

Career Avenues

Graduates of the BCA program have a wide array of career opportunities, including:

1. **Software Developer**: Design, code, and maintain software across various domains such as web, mobile, game, and enterprise applications.
2. **Systems Analyst**: Evaluate and enhance organizational computer systems, design new solutions, and optimize existing processes for improved

efficiency.

3. **Data Scientist**: Analyze large datasets, apply statistical methods and machine learning, and develop predictive models to drive data-informed decisions.

4. **AI Engineer**: Develop and implement AI algorithms, including natural language processing, computer vision, and other intelligent systems.
5. **Cybersecurity Analyst**: Protect systems and data by identifying vulnerabilities, implementing security measures, and responding to security incidents.
6. **Network Engineer**: Design, deploy, and maintain reliable and secure network infrastructures, and troubleshoot network issues to ensure optimal performance.
7. **IT Project Manager**: Oversee technology projects from planning to execution, manage teams, coordinate resources, and ensure timely and budget-compliant delivery.
8. **Database Administrator**: Maintain database systems, ensure data integrity and security, and optimize performance for efficient data management.
9. **Quality Assurance Engineer**: Test and ensure the reliability and functionality of software applications by developing test plans, identifying issues, and collaborating with development teams.
10. **Research and Development**: Engage in cutting-edge research, explore emerging technologies, and contribute to innovation in academia, industry labs, or R&D departments.

Duration

BCA: 3 years/BCA Honors with Research: 4 Years (Full-Time)

Eligibility Criteria

Course Level/Duration/System:

Undergraduate / Three or Four years/6 or 8 Semesters with multiple entry and exit. The following option will be made available to the students joining BCA Research Program:

Multiple Exit Options to Students

Exit Option	Naming convention for the award Certification/Diploma/Degree
Exit after 1st year	▪ **Undergraduate Certificate in Computer Application.**
Exit after 2nd year	▪ **Undergraduate Diploma in Computer Application**
Exit after 3rd year	▪ **Bachelor's in computer application (BCA) with specialization in AI & Data Science**
Exit after 4th year	▪ **Bachelor's in computer application with Honours: BCA(Honours) with specialization in AI & Data Science** **OR** ▪ **Bachelor in Computer Application Honours with Research: BCA (Honours with Research) with specialization in AI & Data Science**

Minimum Eligibility Criteria:

Minimum eligibility criteria for opting the course in the fourth year will be as follows:

1. BCA (Honours with Research): BCA Degree

2. For BCA (Honours): BCA Degree

Note : The students who are eligible for BCA (Honours with Research) shall have choice to pursue istribution either BCA (Honours) or BCA (Honours with Research).

Semester wise credit Distribution

Program Name	I	II	III	IV	V	VI	VII	VIII	Total Credits
BCA (AI & DS)	**24**	**23**	**23**	**25**	**21**	**20**	**19**	**12**	**167**

Student's Structured Learning Experience from Entry to Exit in the Programme

a. **Education Philosophy and Purpose:**

Learn to Earn a Living: At KRMU we believe in equipping students with the skills, knowledge, and qualifications necessary to succeed in the job market and achieve financial stability. All the programmes are tailored to meet industry demands, preparing students to enter specific careers and contributing to economic development.

Learn to Live: The university believes in the holistic development of learners, fostering sensitivity towards society, and promoting a social and emotional understanding of the world. Our aim is to nurture well-rounded individuals who can contribute meaningfully to society, lead fulfilling lives, and engage with the complexities of the human experience.

b. **University Education Objective**

Focus on Employability and Entrepreneurship through Holistic Education using Bloom's Taxonomy. By targeting all levels of Bloom's Taxonomy—remembering, understanding, applying, analysing, evaluating, and creating—students are equipped with the knowledge, skills, and attitudes necessary for the workforce and entrepreneurial success. At KRMU we emphasize on learners critical thinking, problem-solving, and innovation, ensuring application of theoretical knowledge in practical settings. This approach nurtures adaptability, creativity, and ethical decision-making, enabling graduates to excel in diverse professional environments and to innovate in entrepreneurial endeavours, contributing to economic growth and societal well-being.

c. **Importance of Structured Learning Experiences**:

A structured learning experience (SLE) is crucial for effective education as it provides a clear and organized framework for acquiring knowledge and skills. By following a well-defined curriculum, teaching-learning methods and assessment strategies, learners can build on prior knowledge systematically, ensuring that foundational concepts are understood before moving on to more complex topics. This approach not only enhances comprehension but also fosters critical thinking by allowing learners to connect ideas and apply them in various contexts. Moreover, a structured learning experience helps in setting clear goals and benchmarks, enabling both educators and students to track progress and make necessary adjustments. Ultimately, it creates a conducive environment for sustained intellectual growth, encouraging learners to achieve their full potential.

At K.R. Mangalam University SLE is designed as rigorous activities that are integrated into the curriculum and provide students with opportunities for learning in two parts:

Inside the Classroom:

Our educational approach within the classroom is designed to foster **cognitive development** and enhance **student-centric learning**. We prioritize active engagement and deep understanding by employing a variety of methods, tools, and techniques. These include **problem-based learning**, **case studies**, **interactive discussions**, and **technology-enhanced learning platforms**. Our faculty focuses on developing critical thinking, analytical reasoning, and problem-solving abilities, ensuring students achieve well-defined **cognitive outcomes**. Additionally, we integrate the use of **modern teaching tools**, such as Learning Management Systems (LMS), virtual labs, and multimedia resources, to enhance the learning experience and accommodate diverse learning styles. This comprehensive approach not only promotes academic excellence but also nurtures independent learning and lifelong intellectual curiosity.

Outside the Classroom:

Beyond the classroom, our focus shifts to developing students' **people skills** and **psychomotor skills** through hands-on experiences in **industry, community, and laboratory settings**. We encourage participation in internships, industrial visits, community engagement projects, and research opportunities, which allow students to apply theoretical knowledge to real-world challenges. These activities build essential interpersonal skills such as **teamwork, leadership, communication**, and **professional networking**. Simultaneously, students engage in **lab-based learning** and technical workshops that refine their psychomotor abilities, including precision, technical expertise, and problem-solving under practical conditions. Through these outside-the-classroom experiences, students gain a holistic skill set that prepares them to excel in both professional and societal contexts, aligning their education with real-world expectations and industry needs.

d. Educational Planning and Execution

The B.Tech in Computer Science & Engineering (CSE) with Specialization in Cyber Security at K.R. Mangalam University is designed to foster a holistic educational experience, integrating both theoretical knowledge and practical skills. The program offers students a structured path from entry to exit, ensuring they develop technical expertise, problem-solving skills, and professional competencies.

Entry Phase

Upon entering the B.Tech CSE program, students are introduced to the foundational concepts of engineering mathematics, physics/chemistry, and programming. This phase is designed to strengthen their understanding of core scientific and technical principles. Courses such as Engineering Calculus, Fundamentals of Computer Programming using Python, and Basics of Electrical & Electronics Engineering provide a strong foundation. Students also engage in hands-on laboratory sessions to complement theoretical learning, which helps them connect classroom knowledge with real-world applications.

Orientation Program: The university conducts a **one-day orientation program** for first-year students to familiarize them with the university's environment and key aspects. During the program, students are introduced to the university's highlights, important procedures, key functionaries, and the code of conduct. This orientation serves to ensure that students are well-informed and prepared for a smooth transition into university life.

In the first year, students are exposed to critical problem-solving approaches, basic programming, and ethics in engineering, laying the groundwork for their technical and professional growth.

Induction program: The School organizes a **5-day induction program** for first-year students, aimed at providing them with a comprehensive understanding of the school's various aspects. During the program, students are introduced to learning resources, facilities, and opportunities available to them, along with the rules and regulations governing academic and campus life. The induction also includes faculty introductions, guidelines on academic conduct, and detailed information about examination and evaluation methods, ensuring students are well-prepared for their academic journey.

Core Learning

As students advance through the program, they delve deeper into core computer science subjects such as Data Structures, Algorithms, Object-Oriented Programming (C++), Operating Systems, and Database Management Systems. This phase emphasizes both theoretical concepts and their practical application through lab work. The learning is enhanced through exposure to industry-standard tools and techniques, including programming languages like Java and Python, and systems for data management and networking.

The structured academic schedule, with a well-distributed credit system over eight semesters, ensures students acquire deep technical knowledge and skills in software development, systems design, and computing technologies. The Summer Internship Programs and Minor Projects in the curriculum allow students to apply their learning in real-life projects, facilitating experiential learning.

Summer Internships: School offers 2-credit summer internships spanning 6 weeks, where students are encouraged to pursue internships in startups, industries, or premier institutions such as IITs, NITs, and IIITs. In addition, students have the opportunity to earn global certifications during this period. The School also organizes in-house summer schools in collaboration with industry partners, providing further avenues for students to gain hands-on experience and enhance their professional skills. These initiatives are designed to offer students practical exposure, helping them develop industry-relevant expertise.

Value Added Courses: The School offers a range of 2-credit Value Added Courses (VACs) designed to equip students with industry-relevant skills. These courses aim to bridge the gap between academic knowledge and practical application by providing hands-on training that aligns with current industry demands, ensuring that students are well-prepared for professional challenges.

Skill Development

Throughout the program, there is a significant emphasis on developing practical skills and ensuring students are industry-ready. Courses on Artificial Intelligence, Machine Learning, Cloud Computing, and Cybersecurity provide students with cutting-edge knowledge in emerging fields. Value-Added Courses (VAC) like AWS Cloud Fundamentals, Software Testing, Cyber Security, and Design Thinking & Innovation help bridge the gap between academic learning and industry demands.

Collaborative projects, internships, and industry-based certification courses (offered through partnerships with organizations like IBM and Samatrix) further develop students' practical and professional skills, preparing them to thrive in a dynamic workplace.

Capstone and Exit Phase

In the final semesters, students undertake discipline-specific electives and capstone projects. These projects integrate the knowledge and skills they have acquired over the course of their studies. Electives such as Natural Language Processing,

Generative AI, and Blockchain Technologies offer students the flexibility to specialize in areas of their interest.

The final Industrial Project or R&D Project in the eighth semester is a full-time engagement where students work on live industry problems, research projects, or start-up ideas. This project phase, combined with career readiness boot camps and placement preparation activities, ensures that students are equipped to enter the workforce with both technical competence and professional acumen.

Co-Curricular and Extra-Curricular Activities

Students are encouraged to participate in various clubs, societies, and extra-curricular activities. Engagement in activities such as hackathons, coding competitions, and leadership roles in clubs fosters teamwork, leadership, and creativity. These activities complement academic learning, contributing to the students' holistic development.

Community Connect

Aligning with the NEP 2020's vision of social responsibility, the B.Tech CSE program includes community engagement through activities like Extension Projects and social service initiatives. Students work on community projects and participate in programs aimed at addressing local and national challenges, promoting civic responsibility, and developing empathy towards society.

Ethics and Professional Values

The program places a strong emphasis on ethics and professionalism. Students are taught to incorporate ethical considerations in technological development and decision-making processes. This prepares them to not only be skilled engineers but also responsible professionals who contribute positively to society.

Career Counselling and Entrepreneurship

The university offers comprehensive **career counselling services**, providing students with expert guidance on **job placements, internships**, and **skill**

development to help them effectively navigate their career paths. In addition, the university's **incubation center** plays a pivotal role in nurturing **entrepreneurial and leadership skills**, empowering students to explore innovative ideas and launch their own ventures. These initiatives are designed to equip students with the tools and resources necessary for professional success and entrepreneurial growth.

Course Registration

- Every student has to register at the beginning of each semester for the courses offered in the given semester. Major courses are registered centrally for the students. However, for other multidisciplinary courses (DSE, VAC, OE) the students have to register by themselves through ERP.

e. Student Support Services

Mentor-Mentee: At K.R. Mangalam University, the **Mentor-Mentee Program** plays a crucial role in fostering academic and personal growth. Each student is assigned a faculty mentor who serves as a guide throughout their academic journey. This program ensures continuous interaction, where mentors assist students with academic planning, help in resolving personal issues, and provide career guidance. The mentor-mentee relationship transcends the classroom and often involves personal development, professional growth, and overall well-being. The program aims to nurture a supportive environment that enhances the learning experience and helps students reach their full potential.

Counselling and Wellness Services: The university places a strong emphasis on the mental and emotional well-being of its students through its Counselling and Wellness Services. A dedicated team of trained counselors provides personalized sessions, workshops, and wellness programs to address the mental health needs of

the student community. These services focus on holistic well-being, including stress management, emotional resilience, and coping strategies. Regular wellness programs, meditation sessions, and mental health awareness campaigns are conducted to promote a balanced lifestyle and ensure that students can focus on their studies while maintaining their emotional health.

Evaluation of Learning:

At K.R. Mangalam University, assessment and evaluation are integral components of the teaching-learning process, designed to ensure continuous academic progress and holistic development of students. The university follows a Learning Outcome-Based Framework (LOCF), where assessments are aligned with the specific learning outcomes of each program. A variety of assessment methods, including assignments, presentations, quizzes, practical examinations, and project work, are used to gauge students' understanding. The examination system is 100% automated, ensuring timely and transparent evaluation processes. Results are processed efficiently, typically within 13 days, and complaints related to evaluation are minimal, reflecting the university's commitment to maintaining a high standard of academic integrity. This robust system of continuous assessment and feedback fosters a culture of academic excellence and skill development among students.

I. Evaluation Scheme (Theory Courses):

Evaluation Components	Weightage
A. Internal Assessments: **1. Continuous Assessment (30 Marks)** **(Minimum 5 components to be used and to be evenly spaced)** Project/ Quizzes/Test/ Assignments and Essays/	**30 Marks**

Presentations/ Class Participation/ Case Studies/ Reflective Journals	
2. Mid Term Exam	**20 Marks**
B. External Assessments: End term Examination	**50 Marks**
Total	**100 Marks**

II. Evaluation Scheme (Laboratory/Practical Courses):

Evaluation Components	Weightage
Internal Assessments–	
1. Conduct of Experiment	10 Marks
2. Lab Records	10 Marks
3. Lab Participation	10 Marks
4. Lab Project	20 Marks
External Assessments- End term Practical Exam and Viva Voce	50 Marks
Total	**100 Marks**

f. **Feedback and Continuous Improvement Mechanisms:**

K.R. Mangalam University is deeply committed to academic excellence through a robust **feedback and continuous improvement system**. This system is designed to gather comprehensive input from a diverse range of stakeholders, including **students, faculty, alumni, employers, and academic peers**. Feedback is systematically collected and thoroughly analyzed to identify areas for enhancement in **curricula, teaching methodologies, and academic processes**. Based on the insights gained, actionable measures are formulated and communicated to the appropriate bodies for timely implementation.

This structured feedback mechanism ensures that the university's programs remain aligned with **industry trends and societal needs**, providing students with a cutting-edge education that prepares them for real-world challenges. Moreover, the university demonstrates its commitment to continuous improvement through **regular curriculum updates** and the integration of **innovative teaching strategies**, fostering an environment where both faculty and students can grow and excel. By maintaining this cycle of feedback and improvement, K.R. Mangalam University ensures the continuous advancement of its academic offerings and the overall learning experience.

g. **Academic Integrity and Ethics:**

K.R. Mangalam University upholds the highest standards of academic integrity and ethics as a core value of its educational philosophy. The university implements a zero-tolerance policy towards academic misconduct, including plagiarism and other unethical practices. To ensure transparency and honesty in academic work, plagiarism detection software like Drillbit is used to maintain the originality of student submissions and research outputs. Students and faculty are regularly sensitized on the importance of ethical behavior through workshops, seminars, and classroom discussions. The university also integrates ethics and professional values into its curriculum across various disciplines, ensuring that graduates not only excel academically but also demonstrate integrity and responsibility in their professional and personal lives.

Program Structure & Evaluation Scheme

Semester I

SN	Category	Course Code	Course Title	L	T	P	C
1	Major-1	ENCA101	Web Designing Using HTML,CSS, Java Script & PHP	4	0	-	4
2	Major-2	ENMA103	Basics of Mathematics	3	1	-	4
3	IDC-1	ENSP101	Clean Coding with Python	4	0	0	4
4	Major-3	ENCA103	Essentials of Software Engineering	3	1	0	4
4	Major-4	ENCA151	Web Design Lab	-	-	2	1
5	IDC-2	ENSP151	Clean Coding with Python Lab	0	0	2	1
6	VAC-I	VAC-151	Environmental Studies & Disaster Management	2	-	-	2
7	Major-5	ENCA153	Essentials of Software Engineering Lab	0	0	1	2
8	SEC-1	SEC037	Data Visualization using Power BI	0	0	4	2
TOTAL				16	2	9	24

Semester II

SN	Category	Course Code	Course Title	L	T	P	C
1	Major-6	ENCA102	Fundamentals of Object Oriented Programming using C++	3	1	-	4
2	Major-7	ENCA104	Discrete Structure	3	1	-	4
3	IDC-3	ENSP102	Overview of AI, Data Science, Ethics and Foundation of Data Analysis	4	0	0	4
4	SEC-2	SEC039	R Programming for Data Science and Data Analytics Lab	0	0	4	2
5	Major-8	ENCA152	Object Oriented Programming Lab using C++	-	-	2	1
6	IDC-4	ENSP152	Overview of AI, Data Science, Ethics and Foundation of Data Analysis Lab	0	0	2	1
7	Open Elective-1		Students can choose any one elective from the open pool of the university	3	-	-	3
9	Proj-1	ENSI152	Minor Project-I	-	-	-	2
10	MOOC-1		MOOC in the domain of AI & Data Science (Swayam/ NPTEL/AICTE's ELIS)	-	-	-	**2**
TOTAL				13	2	8	23

Semester III

SNo	Category	Course Code	Course Title	L	T	P	C
1	Major-9	ENCA201	Fundamentals of Data Structures	3	1	0	4
2	Major-10	ENCA203	Fundamentals of Java Programming	3	1	0	4
3	IDC-5	ENSP205	Probabilistic Modelling and Reasoning	0	0	4	2
4	Major-11	ENCA251	Fundamentals of Java Programming Lab	0	0	2	1
5	Major-12	ENCA253	Fundamentals of Data Structures Lab	0	0	2	1
6	Open Elective-2		Students can choose any one elective from the open pool of the university	3	0	0	3
7	VAC-2		Students can choose any one elective from the pool of the VAC Courses offered by School	2	0	0	2
8	**AEC-1**	**AEC006**	**Verbal Ability**	**3**	**0**	**0**	**3**
9	INT-1		Summer Internship-I	0	0	0	2
11	AUDIT-1		Competitive Coding - I	2	-	-	0
12	**CS-1**	**CS002**	**Community Service**	**1**	**-**	**-**	**1**
TOTAL				17	2	8	23

*VAC-2					
CODE	**COURSE TITLE**	**L**	**T**	**P**	**C**
VAC170	Design thinking & Innovations for Engineers	-	-	-	2
VAC171	AWS Cloud Fundamentals	-	-	-	2
VAC172	Web Development with open source Frameworks	-	-	-	2
VAC173	Google Data Analytics	-	-	-	2
VAC174	Software Testing using Open Source Frameworks	-	-	-	2
VAC175	Database Management with Open Source Frameworks	-	-	-	2
VAC176	Cyber Security with Open source Frameworks	-	-	-	2
VAC185	Practical Robotics and UAV Applications	2	-	-	2
VAC186	Applied Automotive Engineering: Hands-On Practices and Innovations	2	-	-	2
VAC187	Practical Research Methodology for Engineers	2	-	-	2

Semester IV

SNo	Category	Course Code	Course Title	L	T	P	C
1	Major-13	ENCA202	Fundamentals of Operating Systems	3	1	-	4
2	Major-14	ENCA204	Fundamentals of Database Management Systems	3	1	0	4
3	IDC-6	ENSP212	Foundation of Machine Learning	4	0	0	4
4	Major-15	ENCA252	Fundamentals of Operating System Lab	-	-	2	1
5	IDC-7	ENSP262	Foundation of Machine Learning Lab	0	0	2	1
6	Major-16	ENCA254	Fundamentals of Database Management Systems Lab	0	0	2	1
7	IDC-8	ENSP359	Big Data Analysis with Scala and Spark Lab	-	-	4	2
8	Open Elective-3		Students can choose any one elective from the open pool of the university	3	0	0	3
9	AEC-2	**AEC007**	**Communication & Personality Development**	3	0	0	3
10	PROJ-2	ENSI252	Minor project-II	0	0	0	2
11	AUDIT-2		Competitive Coding - II	2	-	-	0
TOTAL				18	2	10	25

Semester V

SN	Category	Course Code	Course Title	L	T	P	C
1	Major-17	ENCA301	Design and Analysis of Algorithms	3	1	-	4
2	Major-18	ENCA302	Introduction to Computer Organization & Architecture	3	1	0	4
3	IDC-9	ENSP302	Introduction to Natural Language Processing	4	-	-	4
4	SEC-3	SEC040	Data Science - Tools and Techniques Lab	0	0	4	2
5	AEC-3	AEC008	Arithmetic and Reasoning Skills	3	0	0	3
7	Major-19	ENCA351	Design & Analysis of Algorithms Lab	-	-	2	1
8	IDC-10	ENSP352	Natural Language Processing Lab	-	-	2	1
9	INT-2	ENSI251	Summer Internship-II	-	-	-	2
10	AUDIT-3		Competitive Coding - III	2	-	-	0
TOTAL				15	2	8	21

Semester VI

SN	Category	Course Code	Course Title	L	T	P	C
1	DSE-1		Discipline Specific Elective I	4	-	-	4
2	Major-20	ENCA304	Introduction to Computer Networks	3	1	-	4
3	Major-21	ENCA306	Basics of Neural Networks and Deep Learning	4	-	-	4
4	DSE-2		Discipline Specific Elective I Lab	-	-	2	1
5	Major-22	ENCA352	Computer Networks Lab	-	-	2	1
6	Major-23	ENCA354	Neural Networks and Deep Learning Lab	-	-	2	1
7	MOOC-1		Applied Programming and Problem-Solving Skills for Campus Interviews (Infosys Connect Program	-	-	-	2
8	Proj-3	ENSI352	Minor Project-III				2
9	CS-2	CS001	Club/Society	1	-	-	1
TOTAL				12	1	6	20

Semester VII

Discipline Specific Elective - I (Artifical Intelligence)								
(ii)	DSE	ENSP304	Image Processing & Computer Vision	4	-	-	4	
	DSE	ENSP354	Image Processing & Computer Vision lab	-	-	2	1	
(iii)	DSE	ENSP306	Introduction to Generative AI	4	-	-	4	
	DSE	ENSP356	Generative AI lab	-	-	2	1	
(iii)	DSE	ENSP308	Transfer Learning	4	-	-	4	
	DSE	ENSP358	Transfer Learning lab	-	-	2	1	

SN	Category	Course Code	Course Title	L	T	P	C
1	DSE-3		Discipline Specific Elective -II	4	-	-	4
2	DSE-4		Discipline Specific Elective -III	4	-	-	4
2	DSE-5		Discipline Specific Elective -IV	4	-	-	4
3	DSE-6		Discipline Specific Elective -II Lab	-	-	2	1
4	DSE-7		Discipline Specific Elective III Lab	-	-	2	1
5	DSE-8		Discipline Specific Elective IV Lab	-	-	2	1
6	INT-3	ENSI451	Summer Internship-III	2	-	-	2

7	MOOC-2		MOOC in the domain of AI & Data Science (Swayam/ NPTEL/AICTE's ELIS)	-	-	-	2
	TOTAL			14	0	6	19
Discipline Specific Elective - II (Cloud Computing)							
(i)	DSE	ENSP401	Computational Services in The Cloud	4	-	1	4
	DSE	ENSP451	Computational Services in The Cloud Lab	-	-	2	1
(ii)	DSE	ENSP403	Microsoft Azure Cloud Fundamentals	4	-	1	4
	DSE	ENSP453	Microsoft Azure Cloud Fundamentals Lab	-	-	2	1
(iii)	DSE	ENSP405	Storage and Databases on Cloud	4	-	1	4
	DSE	ENSP455	Storage and Databases on Cloud Lab	-	-	2	1
(iv)	DSE	ENSP407	Application Development and DevOps on Cloud	4	-	1	4
	DSE	ENSP457	Application Development and DevOps on Cloud Lab	-	-	2	1
Discipline Specific Elective - III (Full Stack Development)							
(i)	DSE	ENSP409	Mobile Application Development using iOS	4	-	-	4
	DSE	ENSP459	Mobile Application Development using iOS Lab	-	-	2	1
(ii)	DSE	ENSP411	DevOps & Automation	4	-	-	4
	DSE	ENSP461	DevOps & Automation Lab	-	-	2	1
(iii)	DSE	ENSP413	.Net FRAMEWORK	4	-	-	4

(iv)	DSE	ENSP463	.Net FRAMEWORK Lab	-	-	2	1
	DSE	ENSP415	New Age Programming languages	4	0	0	4
	DSE	ENSP465	New Age Programming languages Lab	0	0	2	1
Discipline Specific Elective IV(Cyber Security)							
(i)	DSE	ENSP301	Secure Coding and Vulnerabilities	4	-	-	4
	DSE	ENSP351	Secure Coding and Vulnerabilities lab	-	-	2	1
(ii)	DSE	ENSP303	Cyber Crime Investigation & Digital Forensics	4	-	-	4
	DSE	ENSP353	Cyber Crime Investigation & Digital Forensics lab	-	-	2	1
(iii)	DSE	ENSP305	AI in Cyber Security	4	-	-	4
	DSE	ENSP355	AI in Cyber Security Lab	-	-	2	1
(iv)	DSE	ENSP307	Social Media Security	4	-	-	4
	DSE	ENSP357	Social Media Security Lab	-	-	2	1

Subjects through Industry Support

Sem	Type	Code	Subject	L	T	P	C	
I	DSE	ENSP101	Clean Coding with Python	4	0	0	4	IBM
I	DSE	ENSP151	Clean Coding with Python Lab	0	0	2	1	IBM

II	DSE	ENSP102	Overview of AI, Data Science, Ethics and Foundation of Data Analysis	4	0	0	4	Samatrix
II	DSE	ENSP152	Overview of AI, Data Science, Ethics and Foundation of Data Analysis Lab	0	0	2	1	Samatrix
III	DSE	ENSP205	Probabilistic Modelling and Reasoning	-	-	4	2	Samatrix
IV	DSE	ENSP212	Foundation of Machine Learning	4	-	-	4	Samatrix
IV	DSE	ENSP262	Foundation of Machine Learning lab	-	-	2	1	Samatrix
IV	DSE	ENSP359	Big Data Analysis with Scala and Spark Lab	-	-	4	2	IBM
V	DSE	ENSP302	Natural Language Processing	4	-	-	4	Samatrix
V	DSE	ENSP352	Natural Language Processing Lab	-	-	2	1	Samatrix
V	SEC	SEC040	Data Science - Tools and Techniques Lab	0	0	4	2	Samatrix
VI	DSE	ENCA306	Basics of Neural Networks and Deep Learning	4	-	-	4	Samatrix
VI	DSE	ENCA354	Neural Networks and Deep Learning Lab	-	-	2	1	Samatrix
VI	PROJ	ENSI451	Project & Case Studies by Samatrix	-	-	-	2	Samatrix
				20	0	22	33	

Semester VIII

Category	CODE	COURSE TITLE	L	T	P	Credits
Proj-4	ENSI452	Industry Project /Research Project	-	-	-	12
TOTAL						**12**

Semester	I	II	III	IV	V	VI	VII	VIII	Total
BCA (AI & DS) Honors with Research	24	23	23	25	21	20	19	12	167

TOTAL CREDITS=167 for Honors/Honors with Research

Syllabus

Semester: 1

Web Designing Using HTML, CSS, Java Script & PHP

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: **Web Designing Using HTML,CSS, Java Script & PHP**	**Course Code**	**L-T-P**	**Credits**
	ENCA101	4-0-0	4
Type of Course:	Major		
Pre-requisite(s): NA			

Course Perspective: The "Web Designing Using HTML, CSS, JavaScript, and PHP" course covers essential web design and development skills. It begins with HTML, teaching document structure, tags, and text formatting. Next, CSS is introduced, covering selectors, styling text, backgrounds, colors, and fonts. Students then learn JavaScript, focusing on variables, data types, conditional statements, loops, functions, events. The PHP unit

covers server-side scripting, including syntax, variables, form handling, file operations, and database integration using MySQL. The final unit involves a comprehensive web design project. This syllabus equips students with the skills to create visually appealing and interactive websites.

The Course Outcomes (COs): On completion of the course the participants will be:

CO 1	**Understanding** the fundamental concepts and principles of web designing using HTML, CSS, JavaScript, and PHP.
CO 2	**Applying** appropriate strategies and techniques to solve web design problems using HTML, CSS, JavaScript, and PHP
CO 3	**Designing** ideas and concepts effectively through the implementation of HTML, CSS, JavaScript, and PHP in web design projects.
CO 4	**Analyzing** the different components and elements required for effective web design, including HTML structure, CSS styles, JavaScript functions, and PHP scripts..
CO 5	Developing visually appealing and responsive web pages using HTML, CSS, JavaScript, and PHP, considering user experience and accessibility

Course Outline:

Unit Number: 1	**Title: Introduction to HTML (Hypertext Markup Language) and CSS**	**No. of hours: 10**
Content: **Introduction to HTML:** Hypertext Markup Language, Basic structure of HTML documents HTML tags and elements , Text formatting and links , Images **Introduction to CSS**: Cascading Style Sheets, Internal, External and Embedded CSS, selectors Styling text and backgrounds Working with colors and fonts		
Unit Number: 2	**Title: Advanced HTML and CSS**	**No. of hours: 10**
Content: **Advanced HTML and CSS** : HTML forms and input elements, HTML tables, CSS layout techniques, Box model and positioning Responsive design principles, CSS transitions and animations, Working with media (images, audio, and video)		

Unit Number: 3	Title: JavaScript Fundamentals	No. of hours: 10
Content: **JavaScript Fundamentals**: Introduction to JavaScript, JavaScript variables and data types, Java operators, Conditional statements and loops, JS Functions and events, JavaScript Arrays, DOM, Data Validation using JS, Accessing and modifying HTML elements.		
Unit Number: 4	**Title: Dynamic Web Development with PHP**	**No. of hours: 10**
Content: **Dynamic Web Development with PHP**: Introduction to server-side scripting, PHP syntax and variables, PHP data types, working with forms and user input, Handling files and directories, working with databases (MySQL), Database connections and queries, Inserting, updating, and retrieving data		

References

JavaScript and JQuery: Interactive Front-End Web Development" by Jon Duckett B. V. Ramana, Higher Engineering Mathematics, Tata Mc Graw-Hill Publishing Company Ltd., 2008.

Additional Readings

1. Codecademy Web Development Skill Path: https://www.codecademy.com/learn/paths/webdevelopment

2. FreeCodeCamp Responsive Web Design Certification: **https://www.freecodecamp.org/learn/responsive-web-design/**

3. Codecademy Web Development Skill Path: https://www.codecademy.com/learn/paths/webdevelopmentL.

Web Designing Using HTML, CSS, Java Script & PHP Lab

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: **Web Designing Using HTML, CSS, Java Script & PHP Lab**	**Course Code**	**L-T-P**	**Credits**
	ENCA151	0-0-2	2
Type of Course:	Major		
Pre-requisite(s), if any: Familiarity with HTML and CSS			

Defined Course Outcomes

COs	
CO 1	**Understanding** the Interactive Web Elements.
CO 2	**Developing** Problem-solving and Critical Thinking Skills.
CO 3	**Applying** of Web Development Concepts
CO 4	**Implementing** effective Collaboration and Communication Skills.

Proposed Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Creating a Basic HTML Page: Learn to create a simple HTML page with headings, paragraphs, and basic elements.	CO 2
2	CSS styling sheets: Create a web page to differentiate the use of internal, external and Embedded Styling sheets.	CO 1
3	Styling with CSS: Apply CSS styles to enhance the appearance of HTML elements, such as fonts, colors, and backgrounds	CO 1
4	Designing a Web Page: Working with Frames.	CO 1
5	Designing a Web Page: Working with Images and Hyperlinks.	CO 3
6	Create HTML table: Create a Table using HTML and CSS properties like border, outline, and margin.	CO 3
7	Create Lists: Create a webpage using lists in HTML and CSS properties.	CO 2
8	Responsive Web Design: Design a responsive webpage that adjusts its layout based on different screen sizes using media queries.	CO 3
9	Image Gallery: Develop an image gallery using HTML and CSS, with features like thumbnails and light box effects.	CO 1
10	Create Form: Design a form in a web page using HTML & CSS properties.	CO 2
11	Form Validation: Use JavaScript to validate form inputs, such as required fields, email format, and password strength.	CO 3
12	Implementing data types, variables, expression, and operator in JavaScript.	CO 3
13	Use of conditional statements & looping statements in Java Script.	CO 1

14	Write a JavaScript Program to check whether the given positive number is a multiple of 3.	CO 3
15	JavaScript Operators: Write a JavaScript program to calculate multiplication and division of two numbers.	CO 2
16	JavaScript Arrays: Write a JavaScript program to compute the sum of elements of given array of integers.	CO 3
17	Dropdown Menus: Design dropdown menus using HTML, CSS, and JavaScript to provide a hierarchical navigation structure.	CO3
18	Introduction to PHP: Learn the basics of PHP and create a simple PHP script to display dynamic content on a webpage.	CO 4
19	Database Connectivity: Connect PHP with a MySQL database to retrieve and display data on a webpage.	CO 4
20	User Registration and Login System: Implement a user registration and login system using PHP and MySQL.	CO 4
21	Addition Operations: Perform addition operations using PHP and MySQL to manage database records.	CO 4
22	Retrieve/View Operations: Perform Retrieve/View Operations using PHP and MySQL to manage database records.	CO 4
23	Delete Operation: Perform delete operations using PHP and MySQL to manage database records.	CO 3
24	Update Operation: Perform update operations using PHP and MySQL to manage database records.	CO 3
25	Content Management System (CMS): Build a simple CMS using PHP and MySQL to manage website content dynamically.	CO 4
26	Website Deployment: Learn how to deploy a web project on a server, configure domain and hosting settings, and make the website live on the internet.	CO 3, CO 4

Basics of Mathematics

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Basics of Mathematics	ENMA103	3-1-4	4
Type of Course:	Major Course		
Pre-requisite(s), if any: None			

Course Perspective:

The basics of mathematics course plays a crucial role in laying down foundational knowledge essential for students specializing in computer science. Covering topics such as number systems, algebra, discrete mathematics, logic, and Boolean algebra, the course aims to equip students with the mathematical tools necessary for problem-solving in programming, algorithm design, and data analysis. Understanding number systems like binary and hexadecimal is vital for comprehending how computers process information, while discrete mathematics provides the framework for studying algorithms and data structures. Additionally, logic and Boolean algebra are fundamental in designing logical circuits and understanding the principles of computer logic. Overall, the course bridges theoretical mathematical concepts with practical applications in computer science, preparing students to tackle complex computational problems effectively.

COs	Statements
CO 1	**Applying** the concepts of Determinants & Matrices.
CO 2	**Understanding** Sequence and Series and types of progression
CO 3	**Analyzing** the concepts of Differentiation and Integration

CO 4	**Utilizing** the methods of Differentiation and Integration

Course Outline:

Unit Number: 1	**Title: Determinants & Matrices**	**No. of hours: 8**
Content: **Determinants:** Definition, Minors, Co-factors, Properties of Determinants, Applications of determinants in finding area of triangle. **Matrices:** Definition, Types of Matrices, Addition, Subtraction, Scalar Multiplication and Multiplication of Matrices, Adjoint, Inverse, Solution of system of linear equation by Cramer's Rule.		
Unit Number: 2	**Title: Sequence and Series**	**No. of hours: 8**
Content: **Sequence and Series:** Introduction, Sequences, Series, Arithmetic Progression (A.P), Geometric Progression(G.P), Relationship Between A. M. and G.M., Sum to N terms of Special Series, Principle of Mathematical Induction.		
Unit Number: 3	**Title: Differentiation**	**No. of hours: 8**
Content: **Differentiation:** Derivative of a function, Derivatives of sum, differences, product, and quotient of functions, Derivative of polynomial, trigonometric, exponential, logarithmic, inverse trigonometric and implicit functions, Logarithmic Differentiation, Derivatives of functions in parametric forms, Differentiation by substitution..		
Unit Number: 4	**Title: Integration:**	**No. of hours: 8**
Content: **Integration:** Indefinite integrals, Methods of integration: by substitution, by parts, by partial fractions, Integration of algebraic and transcendental functions.		

Text Books

- A Textbook of Mathematics for XI-XII Students, NCERT Publication Vol. I-II.
- Shanti Narayan, Integral calculus, Sultan Chand & Co.
- Shanti Narayan, Differential calculus, Sultan Chand & Company.
- Babu Ram, Engineering Mathematics, Pearson Education.

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Clean Coding with Python	ENSP101	4-0-0	4
Type of Course:	Major Course		
Pre-requisite(s), if any: None			

Course Perspective. "Clean Coding with Python" is designed to teach students the principles and practices of writing clean, maintainable, and efficient Python code. This course covers essential coding standards, best practices, and design patterns that promote readability, simplicity, and scalability in software development.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the fundamental concepts of Python programming, including basic syntax, control structures, and data types.
CO 2	**Applying** Python data structures and functions to solve problems using lists, tuples, sets, and dictionaries, including the use of functional programming techniques**.**
CO 3	**Implementing** error handling, file operations, and advanced Python features such as decorators, generators, and modules for efficient programming.
CO 4	**Utilizing** Python for advanced tasks like web scraping, working with external file formats (Excel, CSV), GUI development, and database integration.

Course Outline:

Unit Number: 1	Title: Introduction to Python	No. of hours: 8
Content: **Python Introduction and Setup:** Command Line Basics, Installation of Python. Text Editor (VS Code, PyCharm, Anaconda) **Python basics and control structures:** Python data types, Numbers, Variables, getting input from the user, Operators, Statements (If, else, elif), Nested statements, Loops and loop control statements (Break, continue and pass), Strings (Indexing, slicing and formatting).		
Unit Number: 2	**Title: Python Data Structure**	**No. of hours: 10**
Content: **Python Data Structures:** Lists, Tuples, Sets, Dictionaries. Methods and Functions: Introduction to functions, def keyword, *args and **kwargs in python, exercise on functions, Lambda expressions, Map and Filter functions.		
Unit Number: 3	**Title: Python Decorators and generators**	**No. of hours: 10**
Content: **Modules and Package:** Installation using pip **Errors and Exception Handling:** Errors, Exceptions, Try and Except Statement, Catching Specific Exception, Try with else, Finally, Keyword, Raising an exception. File Handling using Python.		
Unit Number: 4	**Title: Python advanced modules**	**No. of hours: 10**
Content: **Python advanced modules:** Datetime module, Math and Random module, OS module **Regular Expressions:** re module, Web Scraping using Python: Web Scraping libraries and practical implementation, working with images using python		
Unit Number: 5	**Title: Working with Excel sheets and CSV files**	**No. of hours: 8**
Content : **Python GUI programming:** Tkinter, Adding Widgets, Buttons etc.SQL queries (DDL, DML,		

DCL, TCL) – Joins, Sub-Queries, Constraints and Inbuilt functions (Date, String, Math)Database handling in python using MySQL db, Fetching and Inserting data using MySQL db.

Learning Experiences

Inside Classroom Learning:

1. Interactive Coding Exercises: Students practice writing Python programs in real-time with in-class coding challenges related to loops, data types, and control structures (Unit 1).
2. Live Code Reviews: Students submit code for peer review, receiving constructive feedback on maintaining clean code principles (CO 2).
3. Hands-on File Handling Tasks: Students work on file manipulation tasks, exploring real-world scenarios using CSV, JSON, and regular expressions (CO 3, Unit 3).
4. Mini-Projects: Small projects like building a basic text-based game to apply Python basics and data structures (CO 5, Unit 2).
5. Web Scraping Practice: Implement web scraping in class to extract data from websites using Python libraries (CO 4, Unit 4).
6. Error-Handling Workshops: Collaboratively debug code with exception-handling techniques, promoting troubleshooting skills (CO 3, Unit 3).
7. Function Design Practice: Group exercises in designing clean, efficient functions using lambda, map, and filter functions (Unit 2).

Outside Classroom Learning:

1. Online Coding Competitions: Participation in coding challenges on platforms like LeetCode or HackerRank to practice clean code (CO 1-5).
2. Industry Guest Lectures: Attending talks by professionals about Python's application in cybersecurity and real-world coding practices (Course Perspective).
3. Group Web Scraping Projects: Students collaborate on web scraping projects using real-world data to extract insights from websites (CO 4, Unit 4).
4. Database Connectivity Assignments: Real-time project work with MySQL databases, creating systems to fetch and insert data (Unit 5).
5. Open-Source Contributions: Students contribute to open-source Python projects, adhering to clean coding standards (CO 1-5).

6. Hackathons: Participation in hackathons focused on building Python-based solutions for cybersecurity challenges (CO 5).
7. Documentation Writing: Students practice writing clear, concise documentation for their Python projects, reinforcing clean coding habits (CO 1-5)

Text and Reference Book

1. J. Peterson, A. Silberschatz, and P. Galvin, "Operating System Concepts", Addison Wesley. 2012
2. V. Aho, R. Sethi, and J. D. Ullman, "Compilers: Principles, Techniques and Tools", Addison-Wesley. 2013
3. R. El. Masri and S. B. Navathe, "Fundamentals of Data Base Systems",Benjamin Cummings. 2013

Additional Readings:

R 1. https://www.tutorjoes.in/python_programming_tutorial/

R 2. https://www.udemy.com/course/100-days-of-code/

R 3. https://favtutor.com/blog-details/7-Python-Projects-For-Beginners

R 4. https://github.com/NaviRocker/100-days-of-python

R 5. https://hackr.io/blog/python-projects

Online Learning Resources

1. **Codecademy**
   - Offers interactive Python courses that cover basic to advanced programming concepts, including data types, control structures, functions, and object-oriented programming.
   - Link: Codecademy Python
2. **Python.org**
   - The official Python website provides a comprehensive beginner's guide, documentation, and tutorials to get started with Python programming.
   - Link: Python Beginner's Guide

CLEAN CODING WITH PYTHON LAB

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Clean Coding with Python Lab	ENSP151	0-0-2	1
Type of Course:	Major		

Defined Course Outcomes

COs	Statements
CO 1	**Developing** solutions to simple computational problems using Pythonprograms.
CO 2	**Solving** problems using conditionals and loops in Python. DevelopPython programs by defining functions and calling them.
CO 3	**Implementing** Python lists, tuples and dictionaries for representingcompound data.
CO 4	**Understanding** the Machine Learning Algorithms.

List of Experiments

Ex. No	**Experiment Title**	**Mapped CO/COs**
1	Develop programs to understand the control structures ofpython	CO 1
2	Develop programs to implement list	CO 3
3	Develop programs to implement Dictionary	CO 3
4	Develop programs to implement tuples	CO 3
5	Develop programs to implement function with stress onscoping	CO 2
6	Develop programs to implement classes and objects	CO 3
7	Develop programs to implement exception handling.	CO 1

8	Develop programs to implement linear search and binarysearch.	CO 2
9	Develop programs to implement insertion sort	CO 2
10	Develop programs to implement bubble sort.	CO 2
11	Develop programs to implement quick sort.	CO 2
12	Develop programs to implement heap sort.	CO 2

ESSENTIALS OF SOFTWARE ENGINEERING

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Essentials of Software Engineering	**Course Code**	**L-T-P**	**Credits**
	ENCA103	3-1-0	4
Type of Course:	Major		
Pre-requisite(s), if any: NA			

Course Perspective: This course covers the fundamentals of software engineering, including understanding system requirements, finding appropriate engineering compromises, effective methods of design, coding, and testing, team software development, and the application of engineering tools, Requirements Engineering technique, Development of UML Diagrams, Software Architecture and Design patterns, Software Testing- Black Box and White Box, Developing Test cases using Equivalence and Boundary value partitioning techniques, Software Refactoring.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Identifying** and describe different software development methodologies.
CO 2	**Summarizing** the stages and activities involved in the software development life cycle.
CO 3	**Analyzing** and evaluate software requirements to identify potential risks and constraints.
CO 4	**Evaluating** software documentation and code quality against industry standards.
CO 5	**Designing and developing** a complex software system that meets specified requirements.

Course Outline:

Unit Number: 1	**Title: Introduction to Models and SRS**	**No. of hours: 10**
Content Summary: **Introduction to Software Engineering**: The evolving role of software, changing nature of software, Software Crisis, Software Processes & Characteristics. **Process models**: The Waterfall model, Agile model, Evolutionary process model, Spiral model. **Software Requirements analysis & specifications:** Requirement engineering, requirement elicitation techniques, Requirements analysis using DFD, Requirement documentation, Nature of SRS, Characteristics & organization of SRS.		
Unit Number: 2	**Title: Software Metrics and System Design**	**No. of hours: 10**

Content Summary: Software Metrics: Size Metrics like LOC, Token Count, Function Count, Design Metrics, Data Structure Metrics, Information Flow Metrics. Cost Estimation Models: COCOMO, COCOMO-II. **System Design:** Design Concepts, Design Specification. UML Diagrams for Structure Modeling, UML Diagrams for Behavior Modeling, UML Diagram for Implementation and deployment modeling.		
Unit Number: 3	**Title: Software Quality Models**	**No. of hours: 10**
Content Summary: **Software Reliability:** Importance, Hardware Reliability & Software Reliability, Failure and Faults, Reliability Models, Basic Model, Logarithmic Poisson Model, **Introduction to Software Quality**: Software Quality control, Quality Assurance, Software Quality Models, CMM & ISO 9001.		
Unit Number: 4	**Title: : Software Testing and Maintenance**	**No. of hours: 10**
Content Summary: **Software Testing**: Testing process, Design of test cases, Functional testing, Boundary value analysis, Equivalence class testing, Decision table testing, Cause effect graphing, Structural testing, Path Testing, Data flow, and mutation testing, Unit Testing, Integration and System Testing, Debugging, Alpha & Beta Testing, Testing Tools & Standards. **Software Maintenance**: Management of Maintenance, Maintenance Process, Maintenance Models, Regression Testing, Reverse Engineering, Software Re-engineering		

References

- K. K. Aggarwal & Yogesh Singh, "Software Engineering", New Age International.
- R. S. Pressman, "Software Engineering – A practitioner's approach", McGraw Hill
- W.S. Jawadekar, "Software Engineering – Principles and Practices", McGraw Hill
- Stephen R. Schach, "Classical & Object-Oriented Software Engineering", IRWIN, TMH.
- James Peter, W. Pedrycz, "Software Engineering: An Engineering Approach", John Wiley &
- Sons.
- Sommerville, "Software Engineering", Addison Wesley.
- K. Chandrasehakhar, "Software Engineering & Quality Assurance", BPB

Web References:

☐ Software Metrics:

- Software Engineering Institute (SEI): https://www.sei.cmu.edu/
- IEEE Computer Society: https://www.computer.org/
- ACM Digital Library: https://dl.acm.org/

☐ Cost Estimation Models:

- COCOMO:
  - Official COCOMO website: http://csse.usc.edu/csse/research/COCOMOII/
- COCOMO-II:
  - COCOMO Model Definition Manual: http://csse.usc.edu/csse/research/COCOMOII/cocomo_main.html

☐ System Design:

- Object Management Group (OMG) - Unified Modeling Language (UML): https://www.uml.org/
- Software Architecture Documentation: https://www.softwarearchitecturebook.com/
- Design Patterns:
  - Gang of Four (GoF) Design Patterns: https://refactoring.guru/design-patterns

☐ Unified Modeling Language (UML) and Software Reliability:

- Official OMG UML Specification: https://www.omg.org/spec/UML/
- Software Reliability Engineering (SRE) resources: https://www.researchgate.net/topic/Software-Reliability-Engineering

☐ **Software Testing and Maintenance**:

- International Software Testing Qualifications Board (ISTQB): https://www.istqb.org/
- IEEE Standard for Software Test Documentation: https://ieeexplore.ieee.org/document/1011891
- Software Maintenance:
  - Software Maintenance and Evolution: A Roadmap: https://ieeexplore.ieee.org/document/5460546

ESSENTIALS OF SOFTWARE ENGINEERING LAB

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Essentials of Software Engineering Lab	ENCA153	0-0-2	1
Type of Course:	Major Course		

Proposed Lab Experiments

Defined Course Outcomes

COs	
CO 1	**Applying** software engineering lifecycle models and methodologies to develop and maintain software systems
CO 2	**Designing** software development processes that align with technical

	understanding and meet specified requirements.

CO 3	**Analyzing** software requirements using appropriate modeling techniques and tools
CO 4	**Developing** test case specifications and implement test cases based on given software requirements

Ex. No.	Experiment Title	Mapped CO/COs
1	Write the complete problem statement	CO1
2	Write the software requirement specification document	CO1, CO3
3	Draw the entity relationship diagram	CO2, CO4
4	Draw the data flow diagrams at level 0 and level 1	CO2, CO4
5	Draw a use case diagram	CO2, CO4
6	Draw an activity diagram of all use cases.	CO2, CO3
7	Draw a state chart diagram of all use cases	CO2,CO3
8	Draw a sequence diagram of all use cases	CO2, CO3
9	Draw a collaboration diagram of all use cases	CO2, CO3
10	Assign objects in sequence diagram to classes and make class diagram	CO2, CO3
	Create test cases for the testing of the modules	CO1, CO5

Reference Books/Materials

1. Stephen R. Schach, "Classical & Object-Oriented Software Engineering", IRWIN, TMH.
2. James Peter, W. Pedrycz, "Software Engineering: An Engineering Approach", John Wiley & Sons.
3. I. Sommerville, "Software Engineering", Addison Wesley.
4. K. Chandrasehakhar, "Software Engineering & Quality Assurance", BPB

Data Visualization using Power BI

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Data Visualization using Power BI	**Course Code**	**L-T-P**	**Credits**
	SEC037	0-0-4	2
Type of Course:	SEC-1		
Pre-requisite(s), if any: Basic knowledge of Excel & data numbers			

Course Perspective: The course on Data Visualization using Power BI provides a comprehensive understanding of how to transform raw data into meaningful insights through visual representation. It begins with the fundamentals of data visualization, emphasizing its importance in decision-making and comparing various tools, with a special focus on Power BI. Students learn to navigate the Power BI interface, connect to diverse data sources, and utilize Power Query for data transformation. The course covers creating and customizing a wide range of visualizations, including charts, maps, and tables, to effectively communicate data insights. The course is divided into 4 modules:

a) Foundation to Data Analytics

b) Data Science Processes

c) Power BI Analytics

d) Introduction to Data Manipulation Using Function

e) Advance Function

The Course Outcomes (COs): On completion of the course the participants will be

COs	Statements
CO 1	**Building** data design using power BI and manage and manipulate data to extract useful information and insights.
CO 2	**Applying** functions to manipulate and analyze data.
CO 3	**Understanding** different data science processes, tools and techniques
CO 4	**Outlining** the key concepts of data handling and how data has evolved
CO 5	**Identifying** the key concepts of data visualization using power BI and will be able to understand the dash board.

Course Outline:

Unit Number: 1	**Title: Foundation to Data Analytics**	**No. of hours: 8**
Content Summary: Introduction to Data Analytics: Working with Formula and Functions, Introduction to Power BI & Charts, Logical functions using Excel, Analysing Datawith Excel.		
Unit Number: 2	**Title: Data Science Processes**	**No. of hours: 8**

Content Summary: Six steps of data science processes define research goals, data retrieval, cleansing data, and correct errors as early as possible, integrating – combine data from different sources, transforming data, exploratory data analysis, Data modeling, model and variable selection, model execution, model diagnostic and model comparison, presentation and automation.

Unit No.: 3	**Title: Power BI Analytics**	**No. of hours: 8**
Content Summary: Power BI Analytics, Data Validation & data models, Power Map for visualize data, Power BI-Business Intelligence , Data Analysis using statistical methods, Dashboard designing.		
Unit Number: 4	**Title: Introduction To Data Manipulation Using Function**	**No. of hours: 8**
Content Summary: Heat Map, Tree Map, Smart Chart,Azure Machine learning , Column Chart, Line Chart, Pie,Bar, Area, Scatter Chart, Data Series, Axes ,Chart Sheet , Trendline , Error Bars, Sparklines, Combination Chart, Gauge, Thermometer Chart.		
Unit Number: 5	**Title: Advan Function**	**No. of hours: 8**
Content Summary: Gantt Chart, Pareto Chart etc, Frequency Distribution, Pivot Chart, Slicers,Tables: Structured, References, Table Styles , What-If Analysis: Data Tables, Correlation model |Regression model.		

Text and Reference Book

1. Microsoft Power BI Complete Reference: Bring Your Data to Life with the Powerful Features of Microsoft Power BI Book by Brian Knight, Devin Knight, and Mitchell Pearson.

Semester 2

FUNDAMENTALS OF OBJECT-ORIENTED PROGRAMMING USING C++

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Concepts of Object Oriented Programming using C++	**ENCA102**	3-1-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Basics of C programming			

Course Perspective: This course introduces students to the fundamental concepts of Engineering Physics, bridging the gap between theoretical physics principles and practical engineering applications. Engineering Physics is crucial for understanding and designing new technologies and systems in various engineering fields such as electronics, materials science, and mechanical engineering. Students will explore core topics including mechanics, Optics, Polarization, and modern physics, with a special emphasis on their relevance to real-world engineering problems. The course is divided into 4 modules:

a) Introduction

b) Classes and Objects

c) Inheritance & Polymorphism

d) Strings and exception handling

The Course Outcomes (COs). On completion of the course the participants will be able to:

COs	**Statements**
CO 1	**Understanding** the procedural and object-oriented paradigm with concepts of streams, classes, functions, data and objects.

CO 2	**Analyzing** dynamic memory management techniques using pointers, constructors, destructors, etc
CO 3	**Applying** the concept of function overloading, operator overloading, virtual functions and polymorphism
CO 4	**Classifying** inheritance with the understanding of early and late binding, usage of exception handling, file handling and generic programming.

Course Outline:

Unit Number: 1	**Title: Foundations of Object-Oriented Programming**	**No. of hours: 10**
Content Summary: ▪ **Programming Approaches:** Procedure-Oriented Approach vs. Object-Oriented Approach ▪ **Basic Concepts of Object-Oriented Programming:** Objects and Classes, Principles of OOP: Abstraction, Encapsulation, Inheritance, Polymorphism, Dynamic Binding and Message Passing ▪ **Characteristics of Object-Oriented Languages:** Benefits and features of OOP languages		
Unit Number: 2	**Title: Classes and Objects**	**No. of hours: 10**
Content Summary: ▪ **Abstract Data Types and Classes:** Concept of abstract data types, Objects and classes, attributes, and methods ▪ **C++ Class Declaration:** Declaring classes in C++, State, identity, and behavior of objects ▪ **Objects:** Local Objects and Global Objects, Scope resolution operator ▪ **Functions in C++:** Friend Functions, Inline Functions ▪ **Constructors and Destructors:** Instantiation of objects, Types of		

constructors (default, parameterized, copy), Static Class Data, Array of Objects, Constant member functions and objects ▪ **Memory Management Operators:** New and delete operators for dynamic memory allocation.		
Unit Number: 3	**Title: Inheritance and Polymorphism**	**No. of hours: 10**
Content Summary: ▪ **Inheritance:** Types of inheritance (single, multiple, hierarchical, multilevel, hybrid), Access specifiers: public, private, and protected, Abstract Classes, Ambiguity resolution using scope resolution operator and virtual base class ▪ **Polymorphism:** Types of Polymorphism (compile-time and run-time), Function Overloading, Operator Overloading ▪ **Pointers and Virtual Functions:** Pointer to objects, this pointer, Virtual Functions and pure virtual functions		
Unit Number: 4	**Title: Advanced C++ Features**	**No. of hours: 10**
Content Summary: ▪ **Strings and Streams:** Manipulating strings, Streams and file handling, File streams and string streams ▪ **Operators and Error Handling:** Overloading operators, Error handling during file operations, Formatted I/O ▪ **Exception Handling:** Throwing an exception, The try block, Catching an exception, Exception objects, Exception specifications, Rethrowing an exception, Catching all exceptions		

Learning Experience

Classroom Learning Experience

1. **Interactive Lectures**: Use PPTs and live coding demonstrations to explain key OOP concepts.
2. **Conceptual Understanding**: Cover fundamental topics like classes, objects, inheritance, and polymorphism.

3. **Problem-Solving Sessions**: Conduct in-class exercises focused on implementing OOP principles in C++.
4. **Theory Assignments**: Assign programming problems that reinforce OOP concepts, discussed in class.
5. **Group Work**: Collaborate on projects that require designing and implementing class structures.
6. **Case Studies**: Analyze real-world applications of OOP in software development.
7. **Continuous Feedback**: Implement in-class quizzes and code reviews for ongoing assessment.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects emphasizing OOP design principles in C++.
2. **Lab Projects**: Facilitate hands-on programming tasks that apply OOP concepts to real-world scenarios.
3. **Question Bank**: Provide practice problems and resources for self-assessment.
4. **Online Forums**: Create platforms for students to discuss coding challenges and share solutions.
5. **Self-Study for Case Studies**: Encourage independent research on OOP best practices and design patterns.
6. **Collaborative Projects**: Organize group projects focused on developing software applications using OOP in C++.

Text books:

T1: Robert Lafore, "Object-Oriented Programming in C++", Sams Publishing, 4th Edition, 2004.

T2: E. Balagurusamy, "Object-Oriented Programming with C++", McGraw Hill Education, 6th Edition, 2017**.**

Reference Book

1. Schildt Herbert, "C++: The Complete Reference", Wiley DreamTech, 2005.Parasons, "Object Oriented Programming with C++", BPB Publication, 1999.
2. Steven C. Lawlor, "The Art of Programming Computer Science with C++", Vikas Publication, 2002.

3. Yashwant Kanethkar, “Object Oriented Programming using C++”, BPB, 2004

Additional Readings:

Online Learning

R 1. C++ Documentation on cppreference.com

- A comprehensive reference that includes detailed documentation of C++ syntax, library functions, and features organized by version.
- **Link: cppreference.com**

Concepts of Object Oriented Programming Using C++ Lab

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Concepts of Object Oriented Programming Using C++ Lab	**Course Code**	**L-T-P**	**Credits**
	ENBC154	**0-0-2**	**1**
Type of Course:	**Major**		
Pre-requisite(s), if any:			

Defined Course Outcomes

COs	Statements
CO 1	**Understanding** class object concepts by using C++.
CO 2	**Developing** programs using inheritance and polymorphism.
CO 3	**Demonstrating** the significance of constructors and destructor.
CO 4	**Illustrating** generic classes using template concepts.
CO5	**Implementing** the concept of file handling.

Lab Experiments

Defined Course Outcomes

COs	**Lab tasks**
CO 1	Implement a simple calculator in C++ that can perform basic arithmetic operations such as addition, subtraction, multiplication, and division. The program should prompt the user to enter two numbers and an operator, then display the result. Use appropriate data types and control structures to handle the calculations and validate user inputs.
CO1	Create a C++ program that checks if a given number is a prime number. The program should prompt the user to enter a number, then use control structures and functions to determine if the number is prime. Display an appropriate message indicating the result.
CO1	Write a C++ program to demonstrate pointer arithmetic by creating an array of integers and using pointers to traverse and manipulate the array elements.

	Implement functions to calculate the sum, average, and maximum value of the array using pointer arithmetic.
CO1	Write a C++ program to perform basic matrix operations such as addition, subtraction, and multiplication. Use two-dimensional arrays to represent the matrices and implement functions for each operation. Ensure the program handles matrices of appropriate sizes and displays the results accurately.
CO1	Create a C++ program that generates the Fibonacci sequence up to a specified number of terms. Use a loop and control structures to generate the sequence and store the terms in an array. Display the generated sequence as the output.
CO1	Write a C++ program to find all unique palindromic substrings in a given string. The function should take a string as input and return a set of strings containing all unique palindromic substrings.
CO2	Create a class Rational to represent rational numbers with attributes numerator and denominator, implementing default, parameterized, and copy constructors, methods to add, subtract, multiply, and divide rational numbers, overloading the << and >> operators for input and output, and a friend function to compare two rational numbers.
CO2	Create a class Student with attributes studentID, name, and grades (an array of integers), implementing default, parameterized constructors, and a destructor, methods to calculate the average grade and display student details, using constant member functions to display details, and implementing dynamic memory allocation for the grades array.
CO2	Create a class InventoryItem with attributes itemID, itemName, and quantity, implementing default, parameterized, and copy constructors, methods to add, remove, and display inventory items, overloading the ++ and -- operators to increase and decrease item quantity, and implementing dynamic memory allocation for inventory items.
CO2	Create a class Polynomial to represent a polynomial with dynamic memory allocation for coefficients, implementing default, parameterized constructors, and a destructor, methods to add, subtract, and multiply polynomials, overloading the +, -, and * operators for polynomial operations, and friend functions to input and output polynomials.
CO3	Develop a Vehicle Management System that demonstrates different types of

	inheritance and polymorphism in C++. The system should manage various types of vehicles, including cars, trucks, and motorcycles, and should be able to perform operations such as adding new vehicles, displaying vehicle details, and comparing vehicles.
CO3	Create a base class Account with methods deposit() and withdraw(). Derive classes SavingsAccount and CurrentAccount from Account. Overload the deposit() and withdraw() methods in the derived classes to include additional parameters like interest rate for SavingsAccount and overdraft limit for CurrentAccount.
CO3	Create a class ComplexNumber to represent complex numbers. Implement operator overloading for +, -, *, and / operators to perform arithmetic operations on complex numbers. Use inheritance to extend the class with additional functionality for polar representation.
CO3	Create a base class Vehicle with attributes make and model, and methods start() and stop(). Derive classes Car, Truck, and Motorcycle from Vehicle. Use dynamic memory allocation (new and delete operators) to create and manage objects of these classes. Implement a function to display details of all vehicles.
CO4	Write a C++ program that compresses a string using the counts of repeated characters. For example, the string "aabcccccaaa" would become "a2b1c5a3". If the "compressed" string would not become smaller than the original string, the function should return the original string. Use streams for efficient string manipulation.
CO4	Write a template-based function in C++ to sort an array of any data type using the quicksort algorithm. Ensure the function works with different data types such as integers, floating-point numbers, and strings.
CO4	Create a custom exception class InvalidInputException in C++ to handle invalid inputs. Implement a function that takes user input and throws an InvalidInputException if the input is not valid. Use try, catch, and throw blocks to handle the exception and display an appropriate error message.
CO4	Write a C++ program that reads a text file, processes the text to remove punctuation, convert to lowercase, and count the frequency of each word. Use string streams for text manipulation and file streams for reading and writing files.
CO4	Implement a template-based stack class in C++ that supports basic stack operations such as push, pop, top, and isEmpty. Ensure the class works with

	different data types and includes appropriate exception handling for stack underflow and overflow.

DISCRETE STRUCTURE

Program Name	**Bachelor in Computer Applications (BCA)**		
COURSE NAME: **Discrete structure**	**COURSE CODE**	**T-P**	**EDITS**
	ENCA104	3-1-0	4
TYPE OF COURSE:	Major		
PRE-REQUISITE(S), IF ANY: Basic of Mathematics			

Course Perspective: This course will discuss fundamental concepts and tools in discrete mathematics with emphasis on their applications to computer science. Topics include logic and Boolean circuits, sets, functions, relations, deterministic algorithms and randomized algorithms, analysis techniques based on counting methods and recurrence relations, trees and graphs etc.

The Course Outcomes (COs): On completion of the course the participants will be:

COs	Statements
CO 1	**Applying** set theory concepts, analyze logical expressions, and use mathematical induction in proofs**.**
CO 2	**Understanding** Model relations, understand graph theory basics, and solve problems related to graphs**.**
CO 3	**Analyzing** Count combinatorial objects, explore discrete structures, and apply optimization techniques
CO 4	**Utilizing** number theory concepts and understand cryptographic algorithms.

Course Outline:

Unit Number: 1	Title: Set Theory and Logic	No. of hours: 10

Set Theory

1. **Basic Concepts**:
   - Notations and terminology (union, intersection, complement)
   - Types of sets (finite, infinite, empty, universal)
   - Multisets (elements with multiplicity)
2. **Ordered Pairs and Cartesian Product**:
   - Definition of ordered pairs
   - Properties of Cartesian product
3. **Set Algebra and Proofs**:
   - Set operations (union, intersection, difference)
   - Proofs of set identities (De Morgan's laws, distributive properties)

Logic

1. **Propositional Logic**:
   - Syntax and semantics
   - Truth tables for logical connectives (AND, OR, NOT)
   - Tautologies and contradictions
2. **Predicate Logic**:
   - Quantifiers (universal and existential)
   - Predicate calculus
   - Proofs using mathematical induction

Unit Number: 2	Title: Relations and Graph Theory	No. of hours: 10

Relations

1. **Representation and Properties**:
   - Matrices, graphs, and directed graphs
   - Reflexive, symmetric, and transitive relations
2. **Equivalence Relations and Partitions**:

- Equivalence classes
- Equivalence partitions

3. **Graph Theory Basics**:
   - Definitions (vertices, edges, degree)
   - Types of graphs (simple, directed, weighted)
4. **Graph Algorithms**:
   - Depth-First Search (DFS)
   - Breadth-First Search (BFS)
   - Shortest path algorithms (Dijkstra's)

Unit Number: 3	**Title: Combinatorics and Discrete Structures**	**No. of hours: 10**

Combinatorics

1. **Counting Principles**:
   - Product rule, sum rule
   - Permutations and combinations
   - Binomial coefficients
2. **Inclusion-Exclusion Principle**:
   - Solving problems with overlapping sets

Discrete Structures

1. **Trees and Recurrence Relations**:
   - Tree properties (rooted, binary, spanning trees)
   - Solving linear recurrence relations
2. **Finite State Machines and Regular Languages**:
   - Deterministic Finite Automata (DFA)
   - Regular expressions
   - Regular languages
3. **Formal Languages and Grammars**:
   - Context-free grammars
   - Chomsky hierarchy

Unit Number: 4	**Title: Number Theory and Cryptography**	**No. of hours: 10**

Number Theory

1. **Divisibility and Modular Arithmetic**:
    - Greatest common divisor (GCD)
    - Modular inverses
2. **Congruences and Fermat's Little Theorem**:
    - Solving congruences
    - Applications in cryptography
3. **RSA Encryption**:
    - Key generation
    - Encryption and decryption

Cryptography

1. **Symmetric-Key Cryptography**:
    - Data Encryption Standard (DES)
    - Advanced Encryption Standard (AES)
2. **Public-Key Cryptography**:
    - RSA algorithm
    - Diffie-Hellman key exchange
3. **Digital Signatures and Hash Functions**:
    - Ensuring data integrity
    - Cryptanalysis techniques

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key concepts in discrete mathematics using PPTs and examples.
2. **Conceptual Understanding**: Cover topics like logic, set theory, combinatorics, and graph theory.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on solving discrete math problems.
4. **Theory Assignments**: Assign theoretical problems, with solutions discussed in class.

5. **Group Work**: Collaborate on projects involving applications of discrete mathematics.
6. **Case Studies**: Analyze real-world applications of discrete math in computer science and cryptography.
7. **Continuous Feedback**: Implement quizzes and feedback sessions to assess understanding.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home problems emphasizing discrete math concepts.
2. **Lab Projects**: Facilitate hands-on activities applying discrete math to programming or algorithms.
3. **Question Bank**: Provide practice problems and resources for self-assessment.
4. **Online Forums**: Create platforms for discussing discrete math problems and solutions.
5. **Self-Study for Case Studies**: Encourage independent research on applications of discrete mathematics.
6. **Collaborative Projects**: Organize group projects focused on real-world problems using discrete math.

Textbooks

1. Discrete Mathematics, Seymour Lipschutz and Marc Lipson, Schaum's Outline Series, 3rd Edition, 2009.
2. Discrete Mathematics, Richard Johnsonbaugh, Pearson, 8th Edition, 2017
3. Discrete Mathematics and Its Applications, Kenneth H. Rosen, McGraw Hill

References

R 1. Elements of Discrete Mathematics, C. L Liu, McGraw-Hill Inc, 1985. Applied Combinatorics, Alan Tucker.

R 2. Concrete Mathematics, Ronald Graham, Donald Knuth, and Oren Patashnik, 2nd Edition - Pearson Education Publishers.

OVERVIEW OF AI, DATA SCIENCE, ETHICS

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Overview of AI, Data Science, Ethics	**Course Code**	**L-T-P**	**Credits**
	ENSP102	4-0-0	4
TYPE OF COURSE:	IDC-3		
PRE-REQUISITE: Basic knowledge of Excel.			

Course Perspective: The students will be studying about Introduction to Data Science, Natural Language, Machine generated Data, Graph-based or Network Data, Audio, Image, Video, and Streaming data. Also, six steps of data science processes define research goals, data retrieval, cleansing data, and correct errors as early as possible, integrating – combining data from different sources, transforming data, exploratory data analysis, Data modeling, model and variable selection, presentation, and automation would be taught to the students. Introduction to Machine Learning and Introduction to Data Analytics is also included in the syllabus.

The Course Outcomes (COs): On completion of the course the participants will be:

COs	**Statements**
CO 1	**Recalling** the key concepts of AI and how AI has evolved
CO 2	**Identifying** the key concepts of Machine Learning and will be able to differentiate between key algorithms such as supervised learning and unsupervised learning.
CO 3	**Distinguishing** key Data Science concepts such as structured and unstructured data, SQL, and NoSQL Database

CO 4	**Examining** the process required the successfully execute a Machine Learning or Data Science project.
CO 5	**Applying** the large-scale data using Excel.

Course Outline:

Unit Number: 1	**Title: Introduction to Data Science**	**No. of hours: 8**
Content: **Introduction to Data Science**: Defining Data Science and Big Data, Benefits and Uses of Data Science and Big Data, Facets of Data, Structured Data, Unstructured Data, Natural Language, Machine generated Data, Graph based or Network Data, Audio, Image, Video, Streaming data, Data Science Process, Big data ecosystem and data science, distributed file systems, Distributed programming framework, data integration framework, machine learning framework, No SQL Databases, scheduling tools, benchmarking tools, system deployments.		
Unit Number: 2	**Title: Data Science Processes**	**No. of hours: 8**
Content: **Data Science Processes**: Six steps of data science processes define research goals, data retrieval, cleansing data, and correct errors as early as possible, integrating – combine data from different sources, transforming data, exploratory data analysis, Data modelling, model and variable selection, model execution, model diagnostic and model comparison, presentation and automation		
Unit Number: 3	**Title: Introduction to Machine Learning**	**No. of hours: 8**

Content: **Introduction to Machine Learning**: What is Machine Learning, Learning from Data, History of Machine Learning, Big Data for Machine Learning, Leveraging Machine Learning, Descriptive vs Predictive Analytics, Machine Learning and Statistics, Artificial Intelligence and Machine Learning, Types of Machine Learning – Supervised, Unsupervised, Semi- supervised, Reinforcement Learning, Types of Machine Learning Algorithms, Classification vs Regression Problem, Bayesian, Clustering, Decision Tree, Dimensionality Reduction, Neural Network and Deep Learning, Training machine learning systems		
Unit Number: 4	**Title: Introduction to AI**	**No. of hours: 8**
Content: **Introduction to AI**: What is AI, Turing test, cognitive modeling approach, the law of thoughts, the relational agent approach, the underlying assumptions about intelligence, techniques required to solve AI problems, level of details required to model human intelligence, successfully building an intelligent problem, history of AI		
Unit Number: 4	**Title: Introduction to Data Analytics**	**No. of hours: 4**
Content: **Introduction to Data Analytics:** Working with Formula and Functions, Introduction to Power BI & Charts, Logical functions using Excel, Analysing Data with Excel.		

References

- Artificial Intelligence 3e: A Modern Approach Paperback – By Stuart J Russell & Peter Norvig; Publisher – Pearson
- Artificial Intelligence Third Edition By Kevin Knight, Elaine Rich, B. Nair – McGraw-Hill
- Artificial Intelligence Third Edition By Patrick Henry Winston – Addison- Wesley Publishing Company

Additional Readings Online

References

- Artificial Intelligence Professional Program, https://online.stanford.edu/programs/artificial-intelligence-professional- program

- Artificial Intelligence (AI), https://www.edx.org/course/artificial- intelligence- ai#!
- 21st-Century Teaching & Learning: Data Science, https://online.stanford.edu/courses/xeduc315n-21st-century-teaching- learning-data- science
- Artificial Intelligence: Principles and Techniques, https://online.stanford.edu/courses/xcs221-artificial-intelligence-principles- and- techniques
- Data Visualization, https://online.stanford.edu/courses/cs448b-data- visualization

OVERVIEW OF AI, DATA SCIENCE, ETHICS AND FOUNDATION OF DATA ANALYSIS LAB

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
OVERVIEW OF AI, DATA SCIENCE , ETHICS AND FOUNDATION OF DATA ANALYSIS LAB	**ENSP152**	0-0-2	1
Type of Course:	IDC-4		
Pre-requisite(s), if any: NA			

Proposed Lab Experiments

Defined Course Outcomes

COs	
CO 1	**Understanding** Data Science and Analytics Concepts
CO 2	**Implementing** the various operations over these data structures.
CO 3	**Learning** the basics of Data Science & Analytics
CO 4	**Developing** project based on Data Science and Anaytics

Ex No	Experiment Title	Mapped CO/COs
1	Write a program that uses functions to perform the following operations on singly linked list: i.Creation ii.Insertion iii.Deletion iv.Traversal	CO1
2	Write a program that uses functions to perform the following operations on doubly linked list: i.Creation ii.Insertion iii.Deletion iv.Traversal	CO1, CO3

3	Write a program that uses functions to perform the following operations on circular linked list: i. Creation ii.Insertion iii.Deletion iv.Traversal	CO2, CO4
4	Write a program that implement stack and its operations using: i. Arrays ii. Pointers	CO2, CO4
5	Write a program that implement queue and its operations using: i. Arrays ii Pointers	CO2, CO4
6	Write a program that implements the following sorting methods to sort a given list of integers in ascending order: Bubble sort ii.Selection sort iii.Insertion sort	CO2, CO3
7	Write a program that use both recursive and non-recursive functions to perform the following searching operations for a Key value in a given list of integers: Linear search Binary search	CO2,CO3
8	Write a program to implement the tree traversal methods.	CO2, CO3
9	Write a program to implement the graph traversal methods.	CO2, CO3
10	Program on Comparative Analysis of Matching Algorithms	CO2, CO3
11	Analyzing the Impact of COVID-19 using Data Science: A Comprehensive Case Study	CO1, CO5
12	Program for Enhancing Data Visualization with Conditional Formatting	CO2, CO3
13	Exploring Pivot Tables in Data Science	CO2, CO3
14	Data Visualization with Power Map	CO1, CO5
15	Write a program for Data Science with Power BI	CO2, CO3
16	Write a program for Building Predictive Models in data science	CO2, CO3
17	Analyzing Sales Wallet Transactions using Data Science: Extracting Insights and Driving Business Growth	CO3, CO4
18	Harnessing the Power of Power Query in Data Science: Extract, Transform, and Analyze Data with Efficiency and Precision	CO3, CO4
19	"Exploring Correlation Methods in Data Science: Unveiling Relationships and Patterns in Complex Datasets	CO3, CO4

R PROGRAMMING FOR DATA SCIENCE AND DATA ANALYTICS LAB

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
R Programming for Data Science and data analytics lab	**SEC039**	0-0-4	2
Type of Course:	SEC2		
Pre-requisite(s): Basics of C programming			

Proposed Lab Experiments

Ex No	Experiment Title	Mapped CO/COs
1	Installation of R and Rstudio.	CO1
2	Create Matrix in R and perform following operations	CO1, CO3
3	Create a matrix A and fill with values from 1 to 12	CO2, CO4
4	Create a matrix B and fill with values from 1 to 12	CO2, CO4
5	Find the transpose of matrix A and matrix B	CO2, CO4
6	Find the multiplication of matrix A and matrix B.	CO2, CO3
7	Find the addition of matrix A and matrix B .	CO2,CO3
8	Find the substation of matrix A and matrix B Subsetting a Matrix	CO2, CO3
9	Create a matrix A and fill with values from 4 to 16 Get first2 rows Subset top 2 row and left 2 columns Subset 3 row and 2 column	CO2, CO3
10	Write a R Program to create a list a_list that contains numbers, strings, logical value, and vectors Add names to the list a_list	CO2, CO3

	Add an element at the end of the list a_list	
11	Create a function calc this function will accept three arguments that include two numeric vectors x and y and one character vector type. The character vector type will define the kind s operation, the user wants to perform.	CO1, CO5
12	Write a R program to find the numbers between 1000 and 1100 that satisfy (i ^ 2) %% 11 equals (i ^ 3) %% 17, where ^ is a power operator and %% (modulo operator) returns the remainder of a division.	CO2, CO3
13	Develop a function that can behave differently according to the type of input object.	CO2, CO3
14	Create scatter plot using more than one dataset use various point styles and colors .	CO1, CO5
15	Create multi-period line plot, with mix different line types and create Multi Series Chart with Legend.	CO2, CO3
16	Create bar chart, pie chart and histogram with random data.	CO2, CO3

SEMESTER 3

FUNDAMENTALS OF DATA STRUCTURES

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Fundamentals of Data Structures	**ENCA201**	3-1-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Basics of Computer Programming			

Course Perspective: This course provides a comprehensive introduction to data structures and algorithms, essential components in the field of computer science that are critical for designing efficient software systems. Data structures serve as the building blocks for data management and organization, crucial for implementing effective algorithms that solve real-world computational problems. The course is structured to not only impart theoretical knowledge but also practical skills through hands-on implementation and problem-solving.

The curriculum is meticulously designed to cover a range of topics from basic to advanced data structures, enabling students to understand and apply various data management techniques effectively. It is divided into four well-defined units:

a) Foundations of Data Structures

b) Linear Data Structures

c) Trees and Graphs

d) Advanced Sorting, Searching, and Algorithm Techniques

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements

CO 1	**Understanding** and apply basic and advanced data structures.
CO 2	**Analyzing** and compare various sorting and searching algorithms.
CO 3	**Designing** and utilize algorithms for advanced data manipulation.
CO 4	**Implementing** and evaluate algorithms using hashing and advanced algorithmic techniques.
CO 5	**Developing** applications that integrate data structures with file I/O operations and handle data dynamically.

Course Outline:

Unit Number: 1	**Title: Foundations of Data Structures**	**No. of hours: 9**
Introduction: Abstract Data Type, Elementary Data Organization. **Measuring efficiency of an Algorithm:** Time and Space Complexity Analysis, Asymptotic notations. **Arrays:** Single and Multidimensional Arrays, Representation of Arrays: Row Major Order, and Column Major Order, Application of arrays, Sparse Matrices.		
Unit Number: 2	**Title: Linear Data Structures**	**No. of hours: 11**
Linked lists: Array and Dynamic Implementation of Single Linked Lists, Doubly Linked List, Circularly Linked List, Operations on a Linked List. Insertion, Deletion, Traversal. **Stacks:** Stack operations: Push & Pop, Array and Linked list implementation of Stack, Applications: Prefix and Postfix Expressions, Recursion. **Queues:** Queue operations: Create, Add, Delete, full and empty queues, Array and linked implementation of queues		
Unit Number: 3	**Title: Trees and Graphs**	**No. of hours: 10**

Searching: Sequential search, Binary Search. **Sorting:** Insertion Sort, Selection, Bubble Sort, Quick Sort, Merge Sort, Heap Sort. **Hashing:** Hash Function, Hash Table, Collision Resolution Strategies.		
Unit Number: 4	**Title: Advanced Sorting, Searching, and Algorithm Techniques**	**No. of hours: 10**
Trees: Basic terminology, Binary Trees, Array and linked list implementation, Types of Binary Tree, Extended Binary Trees, Algebraic Expressions, Tree Traversal algorithms: Inorder, Preorder and Postorder, Threaded Binary trees, Search, Addition and deletion of an element in a binary tree. **Graphs:** Representation (Matrix and Linked), Traversals, Shortest path, Topological sort. Dijkstra's Algorithm, Floyd Warshall's Algorithm, Minimum Spanning Tree Algorithms (Kruskal's Algorithm, Prim's Algorithm).		

Classroom Learning Experience

1. **Interactive Lectures:** Introduce key concepts in data structures using PPTs and coding demonstrations.

2. **Conceptual Understanding**: Cover topics like arrays, linked lists, stacks, queues, trees, and graphs.

3. **Problem-Solving Sessions**: Conduct in-class exercises focused on implementing and using various data structures.

4. **Theory Assignments**: Assign theoretical problems that reinforce data structure concepts, discussed in class.

5. **Group Work**: Collaborate on projects that require designing and optimizing data structures.

6. **Case Studies**: Analyze real-world applications of data structures in software development.

7. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding and coding practices.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects that apply data structure concepts to practical problems.

2. **Lab Projects**: Facilitate hands-on programming tasks using data structures in real-world scenarios.

3. **Question Bank**: Provide practice problems and resources for self-assessment on data structures.

4. **Online Forums**: Create platforms for discussing data structure challenges and solutions.

5. **Self-Study for Case Studies**: Encourage independent research on efficient data structure implementations.

6. **Collaborative Projects**: Organize group projects focused on developing applications using various data structures.

Textbooks

1. Seymour Lipschutz, "Data Structures", 2nd Edition, 2015
2. Aaron Tanenbaum, "Data Structures Using C", 2nd edition, 2016
3. Ellis Horowitz and Sartaj Sahni, "Fundamentals of data structures" 2nd edition, 2017
4. Data Structures Using C (2nd. ed.). Reema Thareja. Oxford University Press, Inc., USA. 2018.

References

1. E. Horowitz and S. Sahani, "Fundamentals of Data Structures", Galgotia Book source Pvt. Ltd.

2. Data Structures & Algorithms in Python by John Canning, Alan Broder, Robert Lafore Addison-Wesley Professional ISBN: 9780134855912.

3. "Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein.

4. Problem Solving with Algorithms and Data Structures Using Python" by Brad Miller and David Ranum.

Additional Readings:

Online References for Learning Data Structures

I) **MIT OpenCourseWare - Introduction to Algorithms (6.006)**

a. Free course materials from MIT's undergraduate course on algorithms, which includes data structures. Lectures, assignments, and exams are available online.

b. Link: MIT OpenCourseWare - Introduction to Algorithms

II) **LeetCode - Data Structures**

a. A platform for practicing coding problems. It provides numerous problems related to data structures, complete with solutions and discussions.

b. Link: LeetCode - Data Structures

Fundamentals of Data Structures Lab

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Fundamentals of Data Structures Lab	**Course Code**	**L-T-P**	**Credits**
	ENCA253	0-0-2	1
Type of Course:	Major		
Pre-requisite(s), if any: Basics of Computer Programming			

Proposed Lab Experiments

Defined Course Outcomes

COs	
CO 1	**Analyzing** and evaluate the time and space complexity of algorithms for various scenarios, demonstrating an understanding of asymptotic notations.
CO 2	**Implementing** and manipulate single-dimensional and multi-dimensional arrays, including operations like insertion, deletion, and traversal.
CO 3	**Developing** and perform operations on linked lists (single, doubly, and circularly linked), stacks, and queues using both array and linked list representations.
CO 4	**Designing** and analyze the efficiency of different sorting and searching algorithms, as well as implement and compare advanced data structures like binary search trees, AVL trees, and graph algorithms.

Lab Experiments

S.N	**Experiment Title**	**Mapped CO/COs**
1	Given an array of integers, perform the following operations:	CO1

	reverse the array, find the maximum and minimum elements, and calculate the sum and average of the elements. Implement functions to perform each operation and ensure the time complexity is optimal.	
2	Given an array, rotate the array to the right by k steps, where k is non-negative. Implement the rotation in-place with O(1) extra space.	CO1
3	Write a function to merge two sorted arrays into a single sorted array. The function should handle arrays of different lengths and ensure the final array is sorted.	CO1
4	Given an array containing n distinct numbers taken from 0, 1, 2, ..., n, find the one that is missing from the array. Implement an efficient algorithm with O(n) time complexity.	CO1
5	Find the kth largest element in an unsorted array. Note that it is the kth largest element in sorted order, not the kth distinct element. Implement an efficient algorithm with O(n log n) time complexity.	CO1
6	Given an unsorted array of integers, find the length of the longest consecutive elements sequence. Your algorithm should run in O(n) time complexity.	CO1
7	Suppose an array sorted in ascending order is rotated at some pivot unknown to you beforehand. Write a function to search for a target value in the array. If found, return its index; otherwise, return -1. Your algorithm should run in O(log n) time complexity.	CO1
8	Write a function to move all zeros to the end of an array while maintaining the relative order of the non-zero elements. Implement the function with O(n) time complexity and O(1) extra space.	CO2
9	Write a class to implement a singly linked list with methods to insert an element at the head, insert an element at the tail, delete an element by value, and traverse the list to print all elements.	CO2
10	Using linked lists, write a function to add two polynomials. Each node in the linked list represents a term in the	CO2

	polynomial with its coefficient and exponent. Implement the function to handle polynomials of different degrees.	
11	Implement a doubly linked list with methods to insert an element at the head, insert an element at the tail, delete an element by value, and reverse the list. Ensure that all operations handle edge cases appropriately.	CO2
12	Create a circular linked list with methods to insert an element, delete an element by value, and traverse the list. Ensure that the list maintains its circular nature after each operation.	CO2
13	Implement a stack using a singly linked list with methods for push, pop, and peek operations. Ensure that the stack handles edge cases, such as popping from an empty stack, appropriately.	CO2
14	Write a function to convert an infix expression to a postfix expression using a stack. The function should handle parentheses and operator precedence correctly.	CO2
15	Create a circular queue using an array with methods for enqueue, dequeue, and checking if the queue is empty or full. Ensure that the circular nature of the queue is maintained after each operation.	CO2
16	Given a sorted array that has been rotated at an unknown pivot, write a function to search for a target value in the array. If the target exists, return its index; otherwise, return -1. Implement an efficient algorithm with O(log n) time complexity using binary search.	CO2
17	Given a collection of intervals, merge all overlapping intervals and return an array of the non-overlapping intervals that cover all the intervals in the input. Implement an efficient algorithm with O(n log n) time complexity.	CO2
18	Given a non-empty array of integers, return the k most frequent elements. Implement an efficient algorithm with O(n log k) time complexity.	CO2
19	Given an array of integers that is already sorted in ascending order, find two numbers such that they add up to a specific	CO2

	target number. Return the indices of the two numbers (1-indexed) as an integer array. Implement an algorithm with O(n) time complexity.	
20	Given an array that has been rotated at an unknown pivot, write a function to search for a target value in the array. Implement the solution using binary search with O(log n) time complexity.	CO3
21	Write a function to merge k sorted linked lists and return it as one sorted list. Implement an efficient solution using a min-heap with O(N log k) time complexity, where N is the total number of nodes.	CO3
22	Given a non-empty array of integers, return the k most frequent elements. Implement the solution with O(n log k) time complexity using a min-heap and a hash map.	CO3
23	Given preorder and inorder traversal of a tree, construct the binary tree. Implement an efficient algorithm with O(n) time complexity using a hash map to store the index of elements in the inorder traversal.	CO4
24	Implement Dijkstra's algorithm to find the shortest path from a source vertex to all other vertices in a weighted graph. Use both adjacency matrix and adjacency list representations for the graph. Ensure the algorithm handles negative weights appropriately.	CO4
25	Implement Kruskal's algorithm to find the minimum spanning tree of a graph. Use a union-find data structure to detect cycles and ensure the algorithm runs in O(E log E) time complexity.	CO4

Fundamentals of Java Programming

Program Name	**Bachelor in Computer Applications (BCA)**		
COURSE NAME: **Fundamentals of Java Programming**	**COURSE CODE**	**T-P**	**EDITS**
	ENCA203	3-1-0	4
TYPE OF COURSE:	Major		
PRE-REQUISITE(S), IF ANY: C++ PROGRAMMING			

Course Perspective: This course provides a comprehensive introduction to Java, one of the most popular and widely used programming languages in the world, particularly known for its portability across platforms from mainframe data centers to smartphones. The "Java Programming" course is meticulously designed to introduce students to the core concepts of object-oriented programming using Java, covering everything from basic constructs to advanced programming features. The curriculum is structured to not only impart theoretical knowledge but also to enhance practical skills through extensive lab sessions, thereby preparing students for real-world software development. The course is divided into 4 modules:

a) Introduction to Java and OOP

b) Inheritance and Polymorphism (Abstract Class, Packages, and Interfaces)

c) Exception Handling, Multithreading and Wrapper Class

d) I/O Stream, File Handling, and Collections

The Course Outcomes (COs): On completion of the course the participants will be:

COs	Statements

CO 1	**Applying** Java fundamentals and basic constructs to write Java programs.
CO 2	**Designing** object-oriented solutions using classes, objects, inheritance, and polymorphism.
CO 3	**Utilizing** interfaces and packages for code structure and reusability.
CO 4	**Implementing** error handling with try-catch-finally and custom exceptions.
CO 5	**Designing** multithreaded applications using synchronization.
CO 6	**Performing** file I/O, work with Java Collections Framework, and manipulate data using collections

Course Outline:

Unit Number: 1	**Title: Introduction to Java and OOP**	**No. of hours: 10**
Content: **Introduction to Java** – Features, and Importance, Java Virtual Machine, Byte Code; Keywords, constants, variables and Data Types, Operators and Expressions, Type casting and conversion; **Java Control Structure** - Decision making – if, if-else, if-else-if ladder, nested if, switch-case, Loop – do, while, for, jump statements – break and continue; **Simple Input and Output** - Scanner Class; Arrays Handling - Single and Multi-dimensional, Referencing Arrays Dynamically; **Java Strings**: String class, Creating & Using String Objects, Manipulating Strings, String Immutability & Equality, Passing Strings To & From Methods. **OOP Paradigm**: Features of OOP, Class and Object in Java: Creating Classes and Objects. Defining Data Members and Member Methods, Overloading Member Methods, Static Members, this Keyword. Constructors: default, parameterized and copy constructors.		

Unit Number: 2	Title: Inheritance and Polymorphism (Abstract Class, Packages, and Interfaces)	No. of hours: 10
Content: **Access Specifies, Introduction to Inheritance** – Derived Class and Super class, super Keyword; **Types of inheritance** – simple, multilevel, multilevel, hierarchical, and hybrid; **Polymorphism** – Static (Method overloading), Dynamic (Method Overriding); Final Class and Method, finalize keyword, Garbage Collection; **Abstract Method and Abstract Class**. Interfaces - Defining an Interface, Implementing an Interface;		
Unit Number: 3	**Title: Exception Handling, Multithreading and Wrapper Class**	**No. of hours: 10**
Content: **Exception Handling -** Definition, Dealing with Errors, The Classification of Exceptions, Declaring Checked Exceptions, Throw an Exception, Creating Exception Classes, Catching Exceptions, finally clause; **Multithreaded Programming** - Fundamentals, Java thread model: priorities, synchronization, messaging, thread classes, Runnable interface, inter thread Communication, suspending, resuming, and stopping threads.		
Unit Number: 4	**Title: I/O Stream, File Handling, and Collections**	**No. of hours: 10**
Content: **File Handling**: File Class Methods, Reading from a File, Writing to a File, Buffered I/O, Character Streams, Byte Streams, File Input/Output Stream, FileReader, FileWriter, BufferedWriter, BufferedReader, FileInputStream, FileOutputStream, File Navigation, File Permissions, Directory Operations, File and Directory Attributes **Java Collections Framework**: Introduction to Java Collections Framework		

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key Java concepts using PPTs and live coding demonstrations.
2. **Conceptual Understanding**: Cover topics like object-oriented programming, exception handling, and Java APIs.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on coding challenges and algorithm implementation.
4. **Theory Assignments**: Assign programming problems that reinforce Java concepts, discussed in class.
5. **Group Work**: Collaborate on projects requiring teamwork in Java application development.
6. **Case Studies**: Analyze real-world applications of Java in software development and enterprise solutions.
7. **Continuous Feedback**: Implement quizzes and code reviews to provide ongoing assessment and improvement.
8.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects applying Java programming concepts to practical problems.
2. **Lab Projects**: Facilitate hands-on programming tasks that apply Java in real-world scenarios.
3. **Question Bank**: Provide practice problems and resources for self-assessment in Java.
4. **Online Forums**: Create platforms for discussing Java programming challenges and solutions.
5. **Self-Study for Case Studies**: Encourage independent research on Java best practices and frameworks.
6. **Collaborative Projects**: Organize group projects focused on developing Java applications and systems.

Text Book

1. HERBERT SCHILDT, ―JAVA – THE COMPLETE REFERENCE‖, ORACLE PRESS.
2. CAY S. HORSTMANN, ―CORE JAVA VOLUME – I FUNDAMENTALS‖, PEARSON.

Additional Readings:

Online Learning Resources

1. **Oracle Java Tutorials**

- The official tutorials from Oracle, which owns Java, are a great starting point. These cover the basics and advanced features of Java.
- Link: Oracle Java Tutorials

2. **Codecademy**

- Codecademy offers an interactive Java programming course that includes exercises and projects to help beginners understand Java from scratch.
- Link: Codecademy Java Course

3. **Java Code Geeks**

- A community-driven site that offers free Java tutorials, articles, and examples. It's a valuable resource for practical tips and best practices.
- Link: Java Code Geeks

4. **LeetCode**

- Excellent for practicing Java coding problems, LeetCode helps in enhancing problem-solving skills in Java, which is crucial for technical interviews.
- Link: LeetCode

JAVA PROGRAMMING LAB

Program Name	**Bachelor in Computer Applications (BCA)**		
COURSE NAME: FUNDAMENTALS OF JAVA PROGRAMMING LAB	**COURSE CODE**	**L-T-P**	**EDITS**
	ENCA251	0-0-2	1
TYPE OF COURSE:	Major		
PRE-REQUISITE(S), IF ANY: C PROGRAMMING			

PROPOSED LAB EXPERIMENTS

DEFINED COURSE OUTCOMES

COS	
CO 1	**Demonstrating** the use of primitive data types, type casting, and basic input/output operations in Java.
CO 2	**Implementing** control structures such as conditional statements and loops to perform arithmetic operations and generate sequences.
CO 3	**Create** and manipulate arrays and demonstrate basic inheritance, polymorphism, and class hierarchies in Java applications.
CO 4	**Develop** and test advanced Java applications using multithreading, file handling, collections, and exception handling to solve real-world problems.

Sr. No.	Lab Task	MAPPED CO/COS
1	A Java Application that manages a small library system	**CO1**
2	A Java Application to manage a simple bank account system	**CO1**
3	A Java class hierarchy for a simple educational institution system	**CO2**
4	A system for managing different types of vehicles in a rental service	**CO2**
5	A Java multithreaded application that simulates a banking system	**CO3**
6	A file management system that supports operations such as reading, writing, copying, and navigating files and directories	**CO4**
7	A contact management system that utilizes different data structures like Lists, Sets, Queues, and Maps	**CO4**

Probabilistic Modelling and Reasoning

Program Name	Bachelor in Computer Applications (BCA)		
Course Name: Probabilistic Modeling and Reasoning	**Course Code**	**L-T-P**	**Credits**
	ENSP205	0-0-4	2
Type of Course:	Probabilistic Modeling and Reasoning		
Pre-requisite(s), if any: Basic knowledge of Statistics			

Course Perspective:

The course on Probabilistic Modelling and Reasoning provides a comprehensive foundation in understanding and applying probabilistic methods to model uncertainty and make informed decisions based on incomplete data. Students will explore fundamental concepts of probability theory, Bayesian inference, and graphical models, while learning both exact and approximate inference techniques. Through hands-on tutorials and practical assignments, participants will gain proficiency in using probabilistic programming languages and apply these skills to real-world problems in various domains such as natural language processing and bioinformatics. This course is essential for developing the analytical and computational skills required for careers in data science, machine learning, and artificial intelligence.

The course is divided into 4 modules:

- Introduction to Statistics
- Probability Theory
- Point Estimations
- Test of Statistical Hypothesis and P-values

The Course Outcomes (COs): On completion of the course the participants will be:

COs	Statements
CO 1	**Explaining** the data-gathering techniques
CO 2	**Inspecting** the data using descriptive statistics
CO 3	**Illustrating** the probability and conditional probability concepts
CO 4	**Distinguishing** between various probability distributions and analyze the data following different probability distributions

CO5	**Solving** inferential statistics problems using point and interval estimation techniques. Infer the statistical problems using hypothesis testing and p value

UNIT WISE DETAILS		
Unit Number: 1	**Introduction to Statistics**	**No. of hours: 8**
Content Summary: Introduction to Statistics. Role of statistics in scientific methods, current applications of statistics **Scientific data gathering:** Sampling techniques, scientific studies, observational studies, data management. Displaying data on a single variable (graphical methods, measure of central tendency, measure of spread), displaying the relationship between two or more variables, a measure of association between two or more variables.		
Unit Number: 2	**Probability Theory**	**No. of hours: 8**
Content: **Probability Theory**: Sample space and events, probability, axioms of probability, independent events, conditional probability, Bayes' theorem. **Random Variables:** Discrete and continuous random variables. Probability distribution of discrete random variables, binomial distribution, Poisson distribution. Probability distribution of continuous random variables, The uniform distribution, normal (Gaussian) distribution, exponential distribution, gamma distribution, beta distribution, t-distribution, $\chi^{"}$ distribution. Expectations, variance, and covariance. Probability Inequalities. Bivariate distributions		
Unit Number: 3	**Point Estimations**	**No. of hours: 8**
Content: **Point Estimations:** Methods of finding estimators, method of moments, maximum likelihood estimators, Bayes estimators. Methods of evaluating estimators, mean squared error, best-unbiased estimator, sufficiency and unbiasedness. **Interval Estimations:** Confidence interval of means and proportions, Distribution free confidence interval of percentiles		
Unit Number: 4	**Test of Statistical Hypothesis and P-values**	**No. of hours: 8**
Content: **Test of Statistical Hypothesis and P-values**: Tests about one mean, tests		

of equality of two means, tests about proportions, p- values, likelihood ratio test, Bayesian tests
Bayesian Statistics: Bayesian inference of discrete random variable, Bayesian inference of binomial proportion, comparing Bayesian and frequentist inferences of proportion, comparing Bayesian and frequentist inferences of mean

Univariate Statistics using Python: Mean, Mode. Median, Variance, Standard Deviation, Normal Distribution, t-distribution, interval estimation, Hypothesis Testing, Pearson correlation test, ANOVA F- test
about intelligence, techniques required to solve AI problems, level of details required to model human intelligence, successfully building an intelligent problem, history of AI

References

- Douglas C. Montgomery, (2012), Applied Statistics and Probability for Engineers, 5th Edition, , Wiley India, ISBN: 978-8-126-53719-

Additional Readings: Specify

Online Learning Resources

☐ Coursera - Probabilistic Graphical Models

- This course, offered by Stanford University, covers the basics of probabilistic graphical models, including representation, inference, and learning.
- Link to course

☐ edX - Bayesian Statistics: From Concept to Data Analysis

- Provided by the University of California, Santa Cruz, this course introduces Bayesian statistics, covering probability theory, Bayesian inference, and various applications.
- Link to course

☐ Khan Academy - Statistics and Probability

- A comprehensive resource that covers a wide range of topics in statistics and probability, from basic concepts to more advanced theories.
- Link to course

☐ MIT OpenCourseWare - Introduction to Probability and Statistics

- This MIT course provides lecture notes, assignments, and exams to help students understand the fundamentals of probability and statistics.
- Link to course

☐ Udacity - Introduction to Statistics

- An introductory course that covers the basics of statistics, including data analysis, probability, and inference.
- Link to course

☐ DataCamp - Foundations of Probability in R

- A hands-on course that teaches the fundamentals of probability theory using the R programming language.
- Link to course

☐ YouTube - 3Blue1Brown: Bayes' Theorem

- An intuitive and visual explanation of Bayes' Theorem, provided by the popular educational.
- Link to course

COMPETITIVE CODING BOOTCAMP- I

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **COMPETITIVE CODING -I**	**Course Code**	**L-T-P**	**Credits**
		3-0-0	0
Type of Course:	AUDIT COURSE		
Contact Hours	30		
Version			

Course Outcomes

CO1	**Understanding** problem-solving strategies and techniques relevant to competitive programming
CO2	**Analyzing** the efficiency of algorithms in terms of time and space complexity using asymptotic notations
CO3	**Applying** core programming concepts such as functions, recursion, and dynamic memory allocation to solve computational problems
CO4	**Implementing** solutions for problems involving arrays and strings, utilizing efficient operations and algorithms

Course Outline:

Unit Number: 1	**Title: Foundations of Competitive Programming**	**No. of hours: 8**
Content: Introduction to Competitive Programming Platforms ▪ Overview of major platforms: Codeforces, LeetCode, HackerRank etc. ▪ Setting up accounts and environment for competitive programming. ▪ Solving introductory problems to get familiar with the platforms.		

Problem-Solving Strategies

- Techniques for solving problems
- Greedy Algorithms: Understanding local optimality leading to global solutions.
- Divide and Conquer: Solving problems by breaking them into subproblems (with examples like Merge Sort).
- Brute Force: Iterative approach to solve problems when constraints are small.

Unit Number: 2	**Title:** Time and Space Complexity of Algorithms	**No. of hours: 8**

Content:

Time and Space Complexity:

- Big O Notation: Definition, examples, and practical importance.
- Common Complexities: O(1), O(log n), O(n), O(n log n), O(n^2), etc.
- Impact of time and space complexity on algorithm performance.
- Asymptotic notations
- Best, Average and worst case analysis of Algorithms

Unit Number: 3	**Title: Core Programming Concepts**	**No. of hours: 8**

Content:

Functions: Definition and Declaration, Function Overloading, Recursion and Backtracking

Pointers: Basics of Pointers and References, Pointer Arithmetic, Dynamic Memory Allocation (malloc, free, new, delete)

Files: File I/O Operations (Reading/Writing), File Handling in C++/Java/Python,

Vectors (in C++/ArrayLists in Java): Declaration, Initialization, and Operations, Dynamic Resizing

Unit Number: 4	Title: Arrays and Strings	No. of hours: 6

Content:

Arrays: Operations, Manipulations

Strings: Operations, Substrings, Pattern Matching

Operations on arrays: Insertion, deletion, and traversal.

String operations: Concatenation, substring search.

Key Problems: Rotating arrays, reversing strings, finding longest substrings without repeating characters

Experiment List

Problem Statement	Mapped COs
1. Two Sum: Find two numbers that add up to a specific target.	CO1
2. Best Time to Buy and Sell Stock: Maximize profit from stock prices.	CO1
3. Valid Parentheses: Check if a string contains valid parentheses.	CO1
4. Greedy Algorithm: Jump Game - Can you reach the end of the array?	CO1
5. Divide and Conquer: Merge Sort implementation to sort an array.	CO1
6. Brute Force: Find all subsets of a given set.	CO1
7. Greedy Algorithm: Minimum Number of Platforms Required for Trains	CO1
8. Divide and Conquer: Maximum Subarray (Kadane's Algorithm)	CO1
9. Brute Force: Count number of occurrences of a substring in a string.	CO1
10. Greedy Algorithm: Coin Change Problem (Minimum Coins)	CO1
11. Time Complexity: Check if a number is prime using O(√n) complexity.	CO2
12. Sorting: QuickSort algorithm with O(n log n) complexity.	CO2
13. Big O Notation: Analyze time complexity of an algorithm.	CO2
14. Space Complexity: Fibonacci with O(n) space complexity.	CO2

Problem Statement	**Mapped COs**
15. Time Complexity: Find first duplicate element in an array with O(n) time.	CO2
16. Time Complexity: Search an element in a rotated sorted array in O(log n) time.	CO2
17. Complexity Analysis: Binary Search Tree operations with complexity O(log n).	CO2
18. Analyze best, average, and worst case for Insertion Sort.	CO2
19. Time and Space Complexity: Check the complexity of an algorithm (recurrences).	CO2
20. Time Complexity: Compute factorial recursively with complexity analysis.	CO2
21. Recursion: Generate all permutations of a string.	CO3
22. Dynamic Memory Allocation: Implement a dynamic array (vector) from scratch.	CO3
23. Backtracking: Solve the N-Queens problem using recursion.	CO3
24. Pointers: Swap two numbers using pointers in C++.	CO3
25. File Handling: Read and write data to a file in Python/C++/Java.	CO3
26. Function Overloading: Implement overloaded functions for adding integers and floats.	CO3
27. Dynamic Memory Allocation: Use malloc and free to manage memory in C.	CO3
28. Recursion: Solve Tower of Hanoi using recursion.	CO3
29. Arrays: Rotate an array to the right by k steps.	CO4
30. Strings: Find the longest substring without repeating characters.	CO4

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Introduce competitive coding concepts and strategies using PPTs and coding demos.
2. **Algorithm Workshops**: Cover key algorithms and data structures essential for coding competitions.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on solving competitive coding problems.
4. **Mock Contests**: Organize timed coding contests to simulate competition environments.
5. **Group Discussions**: Facilitate discussions on problem-solving techniques and optimization strategies.
6. **Continuous Feedback**: Implement peer reviews and performance assessments after practice sessions.

Outside Classroom Learning Experience

1. **Practice Assignments**: Assign coding problems from various online platforms for independent practice.
2. **Online Competitions**: Encourage participation in external coding competitions and hackathons.
3. **Question Bank**: Provide a repository of practice problems and resources for self-assessment.
4. **Online Forums**: Create platforms for students to discuss coding challenges and share solutions.
5. **Self-Study Resources**: Recommend books and online courses for further learning on algorithms and data structures.
6. **Collaborative Projects**: Organize group projects to develop coding applications or solve larger problems together.

Textbooks:

- "Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein.

• "Algorithm Design" by Jon Kleinberg and Éva Tardos.

Online Resources:

• LeetCode (https://leetcode.com/)

• HackerRank (https://www.hackerrank.com/)

• GeeksforGeeks (https://www.geeksforgeeks.org/)

List of Suggested Competitive Programming Courses:

1. Algorithms and Data Structures by MIT OpenCourseWare
2. Introduction to Competitive Programming by NPTEL
3. Competitive Programming by HackerRank
4. The Bible of Competitive Programming & Coding Interviews

All students must complete one online course from the suggested programs.

Web References

- https://www.geeksforgeeks.org/competitive-programming-a-complete-guide/
- https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/
- https://github.com/parikshit223933/Coding-Ninjas-Competitive-Programming
- https://www.hackerearth.com/getstarted-competitive-programming/
- https://www.csestack.org/competitive-coding-questions/

References to Interview Questions

- https://www.simplilearn.com/coding-interview-questions-article
- https://www.csestack.org/competitive-coding-questions/
- https://www.geeksforgeeks.org/a-competitive-programmers-interview/
- https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/

Semester-IV

OPERATING SYSTEMS

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: OPERATING SYSTEMS	**Course Code**	**L-T-P**	**Credits**
	ENCS303	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Basics of programming			

Course Perspective: This course provides a comprehensive introduction to the fundamental principles and practices of operating systems. It covers essential concepts such as process management, memory management, file systems, and I/O systems, as well as more advanced topics like distributed operating systems and concurrent systems. Through this course, students will gain a deep understanding of how operating systems function, how they manage hardware resources, and how they provide services to applications. The course also emphasizes practical skills in implementing and managing operating system components and handling challenges such as process synchronization, deadlocks, and system security. By the end of the course, students will be well-equipped to apply these concepts in designing and optimizing operating systems in various computing environments. The course is divided into 4 modules:

a) Introduction to Operating Systems and Process

b) Memory & File Management

c) Process Synchronization, Deadlocks & I/O Systems

d) Distributed Operating Systems & Concurrent Systems

COs	Statements
CO 1	**Understanding** the fundamental concepts of operating systems, including their structure and types.
CO 2	**Analyzing** process scheduling algorithms and their impact on system performance.
CO 3	**Implementing** and manage memory allocation, paging, and virtual memory techniques.
CO 4	**Examining** process synchronization mechanisms and handle deadlocks in an operating system environment.
CO 5	**Developing** distributed operating systems and concurrent systems with a focus on fault tolerance and recovery mechanisms.

The Course Outcomes (COs): On completion of the course the participants will be:

Course Outline:

Unit Number: 1	**Title: Introduction to Operating System, Process and CPU Scheduling**	**No. of hours: 10**
Introduction: Definition, Role, Types of Operating System, Batch Systems, multi programming, time–sharing, parallel, distributed and real-time systems, Operating system structure, Operating system components and services, System calls, System programs, Virtual machines. **Processes:** Process Concept, Process Scheduling, Operation on Processes, Cooperating Processes, Threads. **CPU Scheduling:** Basic Concepts, Scheduling Criteria, Scheduling Algorithms, Multiple Processor Scheduling, Real-Time Scheduling.		
Unit Number:	**Title: Threads, Synchronization, Deadlock and Memory Management**	**No. of hours: 10**

2		
Threads: overview, Benefits of threads, User and kernel threads, Multithreaded Models, Precedence Graph, Fork-Join, Cobegin-Coend construct. **Inter-process Communication and Synchronization**: Background, The Critical-Section Problem, Synchronization Hardware, Semaphores, Classical Problems of Synchronization, Critical Regions, Monitors, Message Passing. **Deadlocks:** System Model, Deadlock Characterization, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, Recovery from Deadlock. **Memory Management:** Background, Logical vs. Physical Address space, swapping, Contiguous allocation, Paging, Segmentation, Segmentation with Paging.		
Unit Number: 3	**Title: Virtual Memory, Device Management and Secondary-Storage Structure**	**No. of hours: 10**
Virtual Memory: Demand Paging and its performance, Page-replacement Algorithms, Allocation of Frames, Thrashing, page size and other Considerations, Demand Segmentation. **Device Management:** Techniques for Device Management, Dedicated Devices, Shared Devices, Virtual Devices, Independent Device Operation, Buffering, Device Allocation Consideration. **Secondary-Storage Structure:** Disk Structure, Disk Scheduling, Disk Management, Swap Space Management, Disk Reliability.		
Unit Number: 4	**Title: File-System Interface, implementation and Security**	**No. of hours: 10**
File-System Interface: File Concept, Access Methods, Directory Structure. **File-System Implementation:** Introduction, File-System Structure, Basic File System, Allocation Methods, Free-Space Management, Directory		

Implementation. **Security:** Security problems, Goals of protection, Access matrix, Authentication, Program threats, System threats, Intrusion detection.

Learning Experiences

- **Interactive Lectures**: Introduce key concepts of operating systems using PPTs and real-world examples.
- **Conceptual Understanding**: Cover topics like process management, memory management, and file systems.
- **Problem-Solving Sessions**: Conduct in-class exercises focused on system calls and scheduling algorithms.
- **Case Studies**: Analyze real-world operating systems and their architectures.
- **Group Work**: Collaborate on projects that involve designing and implementing operating system components.
- **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding of operating system principles.

- **Outside Classroom Learning Experience**
- **Theory Assignments**: Assign take-home projects requiring application of operating system concepts to practical scenarios.
- **Lab Projects**: Facilitate hands-on tasks involving system programming and OS simulations.
- **Question Bank**: Provide practice problems and resources for self-assessment on operating system topics.
- **Online Forums**: Create platforms for discussing operating system challenges and sharing solutions.
- **Self-Study for Case Studies**: Encourage independent research on current trends and technologies in operating systems.
- **Collaborative Projects**: Organize group projects focused on solving real-world problems using operating system concepts.

Textbooks

1. Operating System Concepts, Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, Wiley
2. Modern Operating Systems, Andrew S. Tanenbaum and Herbert Bos, Pearson, 4th Edition, 2014.
3. Operating Systems: Internals and Design Principles, William Stallings, Pearson, 9th Edition, 2017.
4. Operating Systems: Three Easy Pieces, Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau Arpaci-Dusseau Books, 1st Edition, 2018

References

1. MukeshSinghal and N. G. Shivaratri, "Advanced Concepts in Operating Systems", McGrawHill, 2000
2. Abraham Silberschatz, Peter B. Galvin, G. Gagne, "Operating System Concepts", Sixth Addison Wesley Publishing Co., 2003.
3. Andrew S. Tanenbaum, "Modern Operating Systems", Second Edition, Addison Wesley, 2001.
4. Tannenbaum, "Operating Systems", PHI, 4th Edition.

Additional Readings:

Online Learning References :

I) **MIT OpenCourseWare - Operating System Engineering**

a. Advanced course materials from MIT covering various topics in operating system design and implementation.

b. Link: MIT OpenCourseWare - Operating System Engineering

II) **NPTEL - Operating System by IITs**

a. Online course by IITs providing in-depth coverage of operating system principles and practices.

b. Link: NPTEL - Operating System

OPERATING SYSTEM LAB

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: OPERATINGSYSTEMS LAB	**Course Code**	**L-T-P**	**Credits**
	ENCS351	0-0-2	1
Type of Course:	Major		
Pre-requisite(s), if any: Basics of programming			

Defined Course Outcomes

COs	Statements
CO 1	**Implementing** and analyze process creation, management, and CPU scheduling algorithms, demonstrating the ability to simulate an operating system environment.
CO 2	**Developing** and evaluate multithreaded applications, demonstrating synchronization, deadlock handling, and memory management techniques.
CO 3	**Simulating** virtual memory management, device management, and disk scheduling algorithms, showcasing the application of operating system concepts.
CO 4	**Designing** and implement secure file systems, demonstrating file operations, directory management, and access control mechanisms.

List of Experiments

Ex No	Experiment Title	Mapped CO/COs
1	Implement a program that simulates system calls for basic	CO1

	operations such as process creation, file manipulation, and device management. Demonstrate how system calls interact with the operating system components and services.	
2	Develop a process scheduling simulation that demonstrates different CPU scheduling algorithms (FCFS, SJF, Round Robin, Priority Scheduling). Compare the performance of each algorithm based on scheduling criteria such as turnaround time, waiting time, and response time.	CO1
3	Implement a multi-threaded program to demonstrate the benefits of threads over single-threaded processes. Use different multithreading models such as user-level and kernel-level threads and simulate various thread operations.	CO2
4	Design and implement solutions for classical synchronization problems such as the Producer-Consumer problem, Readers-Writers problem, and Dining Philosophers problem using semaphores, critical regions, and monitors.	CO2
5	Create a simulation to detect and handle deadlocks in a system. Implement deadlock prevention, avoidance, and detection algorithms. Demonstrate recovery from deadlock scenarios.	CO2
6	Implement a demand paging system to simulate virtual memory management. Evaluate the performance of different page-replacement algorithms (FIFO, LRU, Optimal) and analyze the effects of thrashing.	CO3
7	Create a simulation for device management that includes buffering, device allocation, and handling dedicated, shared, and virtual devices. Demonstrate techniques for independent device operation and management.	CO3
8	Develop a disk scheduling simulator to compare the performance of different disk scheduling algorithms (FCFS, SSTF, SCAN, C-SCAN). Implement disk management techniques	CO3

	and swap space management.	
9	Implement a file system simulator to demonstrate different file access methods (sequential, direct, indexed). Design a directory structure and simulate file operations such as creation, deletion, reading, and writing.	CO4
10	Develop a program to simulate file system implementation techniques, including different file allocation methods (contiguous, linked, indexed) and free-space management techniques. Implement a basic file system and directory structure.	CO4

Database Management System

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	Course Code	L-T-P	Credits
Database System Management	**ENCS204**	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: **Nil**			

Course Perspective: This course provides a comprehensive introduction to the fundamental concepts and advanced techniques of database management systems (DBMS). It is designed to equip students with the knowledge and skills required to design, implement, and manage databases effectively. The course covers a broad range of topics, including database architecture, data models, SQL, transaction management, concurrency control, database recovery, and security. The course is divided into 4 modules:

a) Introduction

b) Relational Query Languages

c) Transaction Processing and Storage Strategies

d) Advanced Topics and Database Security

The Course Outcomes (COs): On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the fundamental concepts and architecture of database management systems, including data models and ER modeling.
CO 2	**Utilizing** Structured Query Language (SQL) and relational algebra for effective database querying and manipulation.
CO 3	**Applying** database design principles, including normalization and integrity constraints, to develop well-structured databases.
CO 4	**Analyzing** storage structures, transaction processing, concurrency control, and recovery protocols in databases.
CO 5	**Implementing** security measures and explores advanced database concepts such as distributed databases, data warehousing, and data mining.

Course Outline:

Unit Number: 1	**Title: Introduction**	**No. of hours: 12**

Content:

Introduction to DBMS: Overview, benefits, and applications.

Database System Architecture: Schemas, Instances, Data abstraction, data models (network model, relational model, object-oriented data model), Three schema architecture and data independence

Entity-Relationship Model: Entity Types, Entity Sets, Attributes, and Keys, Relationship Types, Relationship Sets, ER diagrams, Naming Conventions, Design issues.

Integrity Constraints: Primary key, foreign key, unique, not null, check constraints.

Unit Number: 2	**Title: Relational Query Languages**	**No. of hours: 8**

Content:

Relational Database Design, Relational query languages, Relational algebra, Tuple and domain relational calculus.

SQL: DDL (Data Definition Language), DML (Data Manipulation Language), DCL (Data Control Language).

Query Processing and Optimization: Evaluation of relational algebra expressions, query equivalence, join strategies, query optimization algorithms.

Database Design: Functional dependencies, normalization (1NF, 2NF, 3NF, BCNF, 4NF), dependency preservation, lossless decomposition.

Unit Number: 3	**Title: Transaction Processing and Storage Strategies**	**No. of hours: 12**

Content:

Transaction Management: ACID properties, transaction states, serializability, conflict and view serializability.

Concurrency Control: Lock-based protocols, timestamp-based protocols, multi-version concurrency control, deadlock handling.

Database Recovery: Recovery concepts, recovery techniques (log-based recovery, shadow paging), checkpoints.

Unit Number: 4	Title: Advanced Topics and Database Security	No. of hours: 8
Content: **Database Security**: Authentication, authorization, access control, DAC (Discretionary Access Control), MAC (Mandatory Access Control), RBAC (Role-Based Access Control). **Intrusion Detection**: Techniques and tools, SQL injection prevention. **Distributed Databases**: Concepts, architecture, data fragmentation, replication, distributed query processing.		

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key concepts in database management using PPTs and case studies.
2. **Conceptual Understanding**: Cover fundamental topics like data modeling, normalization, and SQL.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on database design and query optimization.
4. **Case Studies**: Analyze real-world database systems and their architectures.
5. **Group Work**: Collaborate on projects that involve designing and implementing databases.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding of database concepts.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects requiring the application of database management principles.
2. **Lab Projects**: Facilitate hands-on tasks to create, manipulate, and query databases using DBMS software.
3. **Question Bank**: Provide practice problems and resources for self-assessment on database topics.

4. **Online Forums**: Create platforms for discussing database challenges and sharing solutions.
5. **Self-Study for Case Studies**: Encourage independent research on current trends and technologies in database management.
6. **Collaborative Projects**: Organize group projects focused on developing database solutions for real-world problems.

Text Books

1. R. Elmasri and S.B. Navathe, 2000, Fundamentals of Database Systems, 3rd Ed, AW.
2. C.J. Date, 2000, An Introduction to Database Systems, 7th ED., Addison-Wesley.
3. Database System Concepts", 6th Edition by Abraham Silberschatz, Henry F. Korth, S. Sudarshan, McGraw-Hill.

Additional Readings:

Online Learning Resources for "Database Management Systems"

1. NPTEL-Database Management System
2. **MIT OpenCourseWare - Database Systems (6.830)**

- Advanced course materials from MIT covering database system internals and advanced topics.
- Link: MIT OpenCourseWare - Database Systems

3. **Oracle - Database 2-Day Developer's Guide**

- Official documentation and guide for Oracle database developers.
- Link: Oracle - Database 2-Day Developer's Guide

4. **SQLBolt - Learn SQL with interactive exercises**

- Interactive SQL tutorials and exercises to practice database querying.
- Link: SQLBolt - Learn SQL

DATABASE MANAGEMENT SYSTEMS LAB

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	L-T-P	Credits
Database Management System Lab	**ENCS254**	0-0-2	1
Type of Course:	Major		

Defined Course Outcomes

COs	Statements
CO 1	**Designing and implementing** database schemas using both open-source and commercial DBMS, defining tables, relationships, and integrity constraints
CO 2	**Developing and analyzing** Entity-Relationship diagrams, relational schemas, and enforce normalization techniques to ensure database efficiency and integrity
CO 3	**Understanding** the Write and execute SQL queries for data definition, manipulation, and complex data retrieval, demonstrating proficiency in relational algebra and transaction processing
CO 4	**Implementing** advanced database concepts including indexing, concurrency control, recovery techniques, security features, and distributed database processing

Proposed Lab Experiments

Ex. No	Lab Task	Mapped CO/COs
1	Analyze and document the benefits of using a DBMS over a traditional file system for managing data. Use a case study of a small retail business to highlight the advantages of a DBMS in handling inventory, sales, and customer data.	CO1
2	Design a three-schema architecture for a university management system. Create the internal schema, conceptual schema, and external schema. Illustrate how data independence is achieved and provide examples of each schema with specific details.	CO1
3	Design and implement an ER model for a university course registration system. The system should include entities such as Students, Courses, Professors, and Enrollments. Define relationships, attributes, and keys	CO1
4	Design a relational database schema for an e-commerce platform that manages Customers, Products, Orders, Order Details, and Payments. Define the entities, their attributes, and the relationships between them, ensuring the schema is normalized to at least 3NF. Use this schema to create an ER diagram and specify primary and foreign keys.	CO2
5	Write SQL scripts to create the e-commerce platform database schema using Data Definition Language (DDL). Create tables for Customers, Products, Orders, Order Details, and Payments, and enforce primary keys, foreign keys, unique constraints, not null constraints, and check constraints. Ensure the database structure supports data integrity and consistency	CO2
6	Populating, Querying, and Securing the E-commerce Database using SQL DML, DCL, and Relational Algebra/Calculus	CO2
7	Design and implement a banking transaction management system that demonstrates the ACID properties. The system should handle various transaction states, ensuring serializability and data integrity. Implement features for depositing, withdrawing, and transferring funds, and simulate scenarios to showcase conflict and view serializability.	CO3

Foundation of Machine Learning

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Fundamentals of Machine Learning	**Course Code**	**L-T-P**	**Credits**
	ENSP212	4-0-0	4
Type of Course:	IDC-6		
Pre-requisite(s), if any:			

Course Perspective:

This course introduces the fundamental concepts of Machine Learning (ML) and provides hands-on experience in applying these concepts to solve real-world problems. Students will explore both the theoretical underpinnings and practical applications of machine learning, focusing on supervised and unsupervised learning techniques.

Prerequisites: Basic knowledge of statistics, probability, and programming

The Course Outcomes (COs): On completion of the course the participants will be:

COs	**Statements**	**Bloom's level**
CO1	Understand Core Concepts and Taxonomy of Machine Learning	Cognitive Level: Understand
CO2	Apply Supervised Learning Models	(Cognitive Level: Apply)
CO3	Design and Implement Ensemble and Unsupervised Learning Algorithms	(Cognitive Level: Create)
CO4	Evaluate Machine Learning Models and Ethical Implications	(Cognitive Level: Evaluate)

Course Outline:

Unit Number:1	**Title: Introduction to MachineLearning**	**No. of hours: 8**

Content Summary

Introduction: Introduction to ML, Conventional VS ML approach, Taxonomy (types) of Machine Learning, Design a Learning System, challenges in Machine Learning, Applications.

Machine learning Workflow: Role of Data, Data Preprocessing, wrangling, Data skewness removal (sampling), Model Training, Model Testing and performance metrics

Unit Number:2	Title: Supervised Learning -I	No. of hours: 14

Content Summary:

Linear models for Regression: Simple Linear Regression, Multiple Linear Regression, Assumptions of Linear Regression, Least Squares Estimation, Gradient estimation, Coefficient Interpretation, Goodness-of-Fit Measures, Regularization Techniques, Model Validation, Applications.

Non-Linear Regression Models: Polynomial Regression, lasso & Ridge regression, Exponential Regression, Logarithmic Regression, evaluation metrics

Logistic Regression: Definition and comparison with linear regression, Odds and Log-Odds, Sigmoid Function, Model Estimation, Interpretation of Coefficients, Model Evaluation, Applications

Unit Number:3	Title: Supervised Learning -II	No. of hours: 08

Content Summary:

k-Nearest Neighbors (k-NN) algorithm: Introduction to k-NN, Choosing the 'k' Value, Distance Metrics, Feature Scaling, Handling Missing Values, Weighted k-NN, Algorithm Efficiency and Optimization, Cross-Validation, Applications of k-NN, Pros and Cons of k-NN

Decision Tree Classification Algorithm: Introduction to Decision Trees,Types of Decision Trees
Building a Decision Tree, Splitting Criteria, Tree Pruning, Handling Missing Data, Decision Trees for Regression, Decision Trees for Classification, Overfitting in Decision Trees, Advantages and Disadvantages of Decision Trees, Applications of Decision Trees.

Naive Bayes Algorithm: Introduction to Naive Bayes, Probability Basics, Bayes' Theorem Application, Assumptions of Naive Bayes, Types of Naive Bayes Models, Gaussian Naive Bayes, Multinomial Naive Bayes, Bernoulli Naive Bayes, Feature Likelihood Estimation, Model Training and Prediction, Laplace Smoothing, Model Evaluation Metrics, Applications of Naive Bayes, Advantages and Limitations

Support Vector Machine (SVM): Introduction to SVM, Linear SVM Classification, Non-Linear SVM Classification, Kernel Trick-Linear Kernel, Polynomial Kernel, Radial

Basis Function (RBF) Kernel Sigmoid Kernel M), Soft Margin Classification, Hyperparameters in SVM, SVM Regression, Feature Scaling Importance in SVM, Model Training and Prediction, Model Evaluation Techniques, Applications of SVM		
Unit Number:4	**Ensemble Learning & Unsupervised Algorithms**	**No. of hours: 10**
Content Summary: **Ensemble Learning**: Introduction to Ensemble Learning, Rationale Behind Ensembles, Methods of Combining Models- Bagging (Bootstrap Aggregating), Boosting, Stacking **Bagging Algorithms**: Random Forests, Extra Trees (Extremely Randomized Trees) **Boosting Algorithms**: AdaBoost (Adaptive Boosting): Gradient Boosting, XGBoost (Extreme Gradient Boosting) Practical Applications of Ensemble Methods, Advantages and Limitations of Ensemble Methods **Unsupervised Learning**: Introduction, K-Means Clustering, PCA (Principal Component Analysis)		

Reference Books:

R 1. T. M. Mitchell, Machine Learning (1 ed.), McGraw Hill, 2017. ISBN 978-1259096952.
R 2. E. Alpaydin, Introduction to Machine Learning (4 ed.), Phi, 2020. ISBN 978-8120350786

Additional Readings: Specify

Self-Learning Components:

Online Learning Resources

Introduction to AI & Machine Learning

1. Coursera - Machine Learning by Stanford University

Taught by Andrew Ng, this course provides a broad introduction to machine learning, data mining, and statistical pattern recognition.

Link: Coursera - Machine Learning

Supervised Learning

Udacity - Intro to Machine Learning with PyTorch and TensorFlow

This course focuses on supervised learning techniques using popular frameworks like PyTorch and TensorFlow.

Link: Udacity - Intro to Machine Learning

Unsupervised Learning

Coursera - Unsupervised Machine Learning by Stanford University

Part of the Machine Learning Specialization, this course dives deep into clustering, association rule learning, and dimensionality reduction.

Link: Coursera - Unsupervised Machine Learning

edX - Data Science: Machine Learning by Harvard University

This course covers several unsupervised learning techniques and is part of Harvard's Data Science Professional Certificate.

Link: edX - Data Science: Machine Learning

edX - MLOps (Machine Learning Operations) Fundamentals by Google Cloud

Learn the basics of MLOps and how to deploy, scale, and maintain machine learning models using Google Cloud.

Link: edX - MLOps Fundamentals

GitHub - Awesome Machine Learning

A curated list of awesome machine learning frameworks, libraries, and software.

Link: GitHub - Awesome Machine Learning

Fundamentals of Machine Learning Lab

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Fundamentals of Machine Learning Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP262	**0-0-2**	**1**
Type of Course:	IDC-7		
Pre-requisite(s), if any:			

COs	
CO 1	Explain the use of Machine Learning Models in business and understand machine learning models can be used to solve business problems.
CO 2	Compare machine learning algorithms such as supervised, unsupervised, and reinforcement learning models.
CO 3	Identify the performance of different machine learning models and compare them to optimize the results.
CO 4	Make use continuous and discrete data set to fit regression andclassification models.

Lab Experiments

Defined Course Outcomes

List of Programs

Ex No	Experiment Title	Mapped CO/COs
1	Prediction using simple linear regression	CO1
2	Prediction using multiple linear regression	CO1
3	Classification using Logistics regression	CO1

4	Classification using linear discriminant analysis	CO1
5	Classification using support vector machine.	CO2
6	Classification using Guassian Naïve Bayes	CO2
7	Classification using decision Tree	CO2
8	Classification using Random Forest.	CO1
9	Classification using K nearest neighbour.	CO4
10	Write a program to Retrieve Data for a machine Learning project.	CO3
11	Write a program to Conduct Exploratory Data Analysis using Python	CO3
12	Write a program to Clean the Data using Python	CO4
13	Write a program for Data Modeling using Python	CO4
14	Write a program to implement multiple linear regression.	CO2
15	Write a program to scale the data and implement linear regression using sklearn.	CO2
16	Write a program to implement multiple logistic regression.	CO2
17	Write a program for graphical representation ofdata.	CO1
18	Write a program to implement genetic algorithms.	CO4
19	Write a program to implement CNN.	CO3
20	Write a program to implement LSTM.	CO3

LIFE SKILLS FOR PROFESSIONALS-II

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Life Skills for Professionals - II	**AEC012**	3-0-0	3
Type of Course:	AEC		
Pre-requisite(s), if any: NA			

Course Perspective: This course is designed to equip students with essential life skills that are critical for personal and professional success. "Life Skills for Professionals-II" builds on foundational skills and introduces advanced techniques and practices that enhance personality, arithmetic proficiency, presentation capabilities, and leadership qualities. The course focuses on practical application and real-world scenarios to ensure students can effectively navigate professional environments. The course is divided into 5 modules:

a) Personality Improvement:

b) Ratio & Its Application

c) Arithmetic

d) Presentation Skills

e) Leadership Skills

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the fundamental theories, models, and principles of communication.
CO 2	**Applying** the ability to communicate effectively through spoken and written forms. It includes developing skills in public speaking, interpersonal communication, professional writing, and persuasive

	communication.
CO 3	**Evaluating** the development of teamwork and collaboration skills. It includes activities such as group projects, team-building exercises, and simulations that allow students to practice effective communication and collaboration within diverse teams
CO 4	**Improving** their communication skills in different professional and personal contexts, such as interviews, networking events, customer interactions, and interpersonal relationships
CO 5	**Analyzing** ideas and information clearly and concisely through spoken language. They will develop the ability to articulate their thoughts, use appropriate vocabulary, and convey their message with clarity.

Course Outline:

Unit Number: 1	**Title: Personality Improvement**	**No. of hours: 6**
Content Summary: Asking for and giving information, Offering and responding to offers, Requesting and responding to requests, Congratulating people on their success, Asking questions and responding politely, Apologizing and forgiving		
Unit Number: 2	**Title: Ratio & its application**	**No. of hours: 6**
Content Summary: Time & Work, Time & Distance, Train, Boat & Stream, Permutation & combination, Probability		
Unit Number: 3	**Title: Arithmetic**	**No. of hours: 6**

Content Summary: Inequalities, Log, progression, Mensuration, BODMAS		
Unit Number: 4	**Title: Presentation Skills**	**No. of hours: 6**
Content Summary: Presentation Skills, Telephone etiquettes, LinkedIn Profile and professional networking, Video resumes & Mock interview sessions.		
Unit Number: 5	**Title: Leadership skills**	**No. of hours: 6**
Content Summary: Nurturing future leaders, increasing productivity of the workforce, Imparting Self-leadership, Executive leadership.		

Learning Experiences

1. **Interactive Lectures**: Introduce advanced life skills concepts using PPTs and real-life scenarios.
2. **Conceptual Understanding**: Cover topics like emotional intelligence, conflict resolution, and leadership.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on real-world workplace challenges.
4. **Group Discussions**: Facilitate discussions on ethics, diversity, and effective communication in professional settings.
5. **Guest Speakers**: Invite industry leaders to share insights and experiences related to life skills.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess application of life skills.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign reflective essays on personal development and career aspirations.
2. **Workshops**: Facilitate hands-on sessions for practicing negotiation, networking, and public speaking skills.
3. **Question Bank**: Provide resources for self-assessment on advanced life skills development.
4. **Online Forums**: Create platforms for discussing personal growth and professional challenges.
5. **Self-Study for Case Studies**: Encourage independent research on successful professionals and their life skills.
6. **Collaborative Projects**: Organize group projects focused on community engagement and leadership initiatives.

Textbooks

I) Aggarwal, R. S. (2014). Quantitative aptitude (Revised edition).

II) Gladwell, M. (2021). Talking to strangers.

III) Scott, S. (2004). Fierce conversations.

References

- "The 7 Habits of Highly Effective People" by Stephen R. Covey
- "Presentation Skills: The Essential Guide for Students" by Joan van Emden and Lucinda Becker
- "Emotional Intelligence: Why It Can Matter More Than IQ" by Daniel Goleman
- "Arithmetic for Competitive Examinations" by R.S. Aggarwal
- "The Art of Public Speaking" by Dale Carnegie

MINOR PROJECT-II

Program Name	Bachelor in Computer Applications (BCA)		
Course Name: Minor Project-II	Course Code	L-T-P	Credits
	PROJ2	---	2
Type of Course:	Project		
Pre-requisite(s), if any: NA			

Duration:

The minor project will last for **three** months.

Project Requirements:

1. **Understanding of Societal Problems:**
   - Students must have a basic understanding of societal problems, the concerned domain, and relevant issues.
2. **Critical Thinking and Problem Formulation:**
   - Students are expected to think critically about formulated problems and review existing solutions.
3. **Data Gathering and ETL Activities:**
   - Students should gather relevant data and perform ETL (Extract, Transform, Load) activities to prepare the data for analysis.
4. **Innovation and Entrepreneurship Focus:**
   - Students should develop innovative ideas or entrepreneurial solutions to address the identified problems.
5. **Implementation (Optional):**
   - While implementation of the proposed solutions is encouraged, it is not strictly required. The focus should be on idea development.

Guidelines:

1. **Project Selection:**

- Choose a societal problem relevant to the field of computer science and engineering.
- Ensure the problem is specific and well-defined.

2. **Literature Review:**
   - Conduct a thorough review of existing literature and solutions related to the problem.
   - Identify gaps in existing solutions and potential areas for further investigation.
3. **Data Gathering and ETL:**
   - Collect relevant data from various sources.
   - Perform ETL activities to clean, transform, and load the data for analysis.
4. **Analysis and Critical Thinking:**
   - Analyze the problem critically, considering various perspectives and implications.
   - Evaluate the effectiveness and limitations of current solutions.
5. **Innovation and Idea Development:**
   - Develop innovative ideas or entrepreneurial solutions to address the identified problem.
   - Focus on the feasibility, impact, and potential of the proposed solutions.
6. **Documentation:**
   - Document the entire process, including problem identification, literature review, data gathering, ETL activities, analysis, and ideas.
   - Use appropriate formats and standards for documentation.
7. **Presentation:**
   - Prepare a presentation summarizing the problem, existing solutions, data analysis, and proposed ideas.
   - Ensure the presentation is clear, concise, and well-structured.

Evaluation Criteria for Minor Project (Out of 100 Marks):

1. **Understanding of Societal Problems (15 Marks):**

- Comprehensive understanding of the problem: 15 marks
- Good understanding of the problem: 12 marks
- Basic understanding of the problem: 9 marks

- Poor understanding of the problem: 5 marks
- No understanding of the problem: 0 marks

2. **Critical Thinking and Analysis (20 Marks):**

- Exceptional critical thinking and analysis: 20 marks
- Good critical thinking and analysis: 15 marks
- Moderate critical thinking and analysis: 10 marks
- Basic critical thinking and analysis: 5 marks
- Poor critical thinking and analysis: 0 marks

3. **Data Gathering and ETL Activities (20 Marks):**

- Comprehensive and effective ETL activities: 20 marks
- Good ETL activities: 15 marks
- Moderate ETL activities: 10 marks
- Basic ETL activities: 5 marks
- Poor ETL activities: 0 marks

4. **Innovation and Idea Development (25 Marks):**

- Highly innovative and feasible ideas: 25 marks
- Good innovative ideas: 20 marks
- Moderate innovative ideas: 15 marks
- Basic innovative ideas: 10 marks
- Poor innovative ideas: 5 marks
- No innovative ideas: 0 marks

5. **Documentation Quality (10 Marks):**

- Well-structured and detailed documentation: 10 marks
- Moderately structured documentation: 7 marks
- Poorly structured documentation: 3 marks
- No documentation: 0 marks

6. **Presentation (10 Marks):**

- Clear, concise, and engaging presentation: 10 marks
- Clear but less engaging presentation: 7 marks
- Somewhat clear and engaging presentation: 3 marks

- Unclear and disengaging presentation: 0 marks

Total: 100 Marks

Course Outcomes:

By the end of this course, students will be able to:

1. **Understand Societal Issues:**
   - Demonstrate a basic understanding of societal problems and relevant issues within the concerned domain.
2. **Critical Thinking:**
   - Think critically about formulated problems and existing solutions.
3. **Data Management:**
   - Gather relevant data and perform ETL activities to prepare the data for analysis.
4. **Innovation and Entrepreneurship:**
   - Develop innovative ideas or entrepreneurial solutions to address identified problems.
5. **Literature Review:**
   - Conduct comprehensive literature reviews and identify gaps in existing solutions.
6. **Documentation:**
   - Document findings and analysis in a well-structured and appropriate format.
7. **Presentation Skills:**
   - Present findings and analysis effectively, using clear and concise communication skills.
8. **Problem Analysis:**
   - Analyze problems from various perspectives and evaluate the effectiveness of existing solutions.
9. **Professional Development:**
   - Develop skills in research, analysis, documentation, and presentation, contributing to overall professional growth.

Learning Experiences

Classroom Learning Experience

1. **Project Kickoff**: Introduce project objectives and expectations through orientation sessions.

2. **Research Methodology**: Cover essential techniques for conducting research and project planning.
3. **Problem-Solving Sessions**: Conduct workshops focused on overcoming project-related challenges.
4. **Progress Presentations**: Facilitate sessions for students to present their project updates and receive feedback.
5. **Group Collaboration**: Encourage teamwork to enhance project development and idea exchange.
6. **Continuous Feedback**: Implement peer reviews and mentor check-ins to assess progress and learning.

Outside Classroom Learning Experience

1. **Independent Research**: Assign tasks that require in-depth research and exploration of project topics.
2. **Hands-On Implementation**: Facilitate practical application of project concepts in real-world scenarios.
3. **Documentation**: Encourage students to maintain detailed project logs and documentation.
4. **Online Collaboration Tools**: Create platforms for students to communicate and share resources effectively.
5. **Self-Assessment**: Provide tools for students to evaluate their contributions and project outcomes.
6. **Final Presentation**: Organize sessions for students to present their completed projects to peers and faculty.

COMPETITIVE CODING - II

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
COMPETITIVE CODING -II		3-0-0	0
Type of Course:	AUDIT COURSE		
Contact Hours	30		

Course Outcomes

CO1	**Understanding** fundamental tree structures, including AVL trees, and their balancing mechanisms.
CO2	**Applying** graph representations (adjacency matrix and adjacency list) to solve basic graph traversal problems.
CO3	**Implementing** shortest path algorithms such as Dijkstra's algorithm and Bellman-Ford.
CO4	**Exploring** dynamic programming concepts, including memoization and tabulation, to solve classic problems.

Course Outline:

Unit Number: 1	**Title: Object-Oriented Programming Concepts**	**No. of hours: 8**
Content: **OOP Basics:** Encapsulation, Inheritance, Polymorphism, Class Design and Object Creation **C++ OOP Concepts:** Classes and Objects, Constructors/Destructors, Operator Overloading, Inheritance, Virtual Functions **Java OOP Concepts:** Classes and Objects, Constructors, Method Overloading,		

Inheritance, Polymorphism, Abstract Classes, Interfaces **Python OOP Concepts:** Classes and Objects, Constructors, Method Overloading (via default arguments), Inheritance, Polymorphism, Multiple Inheritance		
Unit Number: 2	**Title:** Linked Lists, Stacks and Queues	**No. of hours: 8**
Content: Linked Lists ▪ Singly and doubly linked lists: Creation, insertion, deletion, traversal. ▪ Key Problems: Reversing a linked list (iterative and recursive), detecting cycles using Floyd's cycle-finding algorithm. Stacks and Queues : ▪ Stack operations: Push, pop, top, isEmpty. ▪ Queue operations: Enqueue, dequeue, front, isEmpty. ▪ Applications: Parentheses matching, queue-based problems (LeetCode challenge - sliding window problems).		
Unit Number: 3	**Title: Sorting & Searching**	**No. of hours: 8**
Content Basic Sorting Algorithms • Implementing Bubble Sort, Selection Sort, Insertion Sort. • Understanding the time complexities and use cases of each algorithm. • Key Problems: Sorting small arrays, finding the median, custom sorting based on conditions (frequent LeetCode challenge). Advanced Sorting Algorithms • Implementing Merge Sort, Quick Sort, Heap Sort. Binary Search • Implementing binary search for sorted arrays. • Applications: Finding an element in a sorted array, finding the position to insert		

an element, LeetCode challenges like searching for ranges.		
Unit Number: 4	**Title: Trees**	**No. of hours: 6**
Content: **Basic Tree Concepts** • Introduction to tree terminology and operations. • Tree Traversals: Preorder, inorder, postorder. • Key Problems: Printing all elements in a tree, finding the depth of a tree. **Binary Trees** • Basic operations on binary trees: Insertion, deletion, searching. • Key Problems: Finding the height of a binary tree, counting leaf nodes, lowest common ancestor (common LeetCode challenges). **Binary Search Trees** Understanding BST properties: Every left subtree is smaller, and every right subtree is larger.		

Learning Experiences:

Classroom Learning Experience

1. **Advanced Lectures**: Introduce complex algorithms and techniques using PPTs and coding demonstrations.
2. **Algorithm Workshops**: Cover advanced topics like dynamic programming, graph algorithms, and optimization strategies.
3. **Intensive Problem-Solving Sessions**: Conduct in-class exercises focused on challenging competitive coding problems.
4. **Mock Contests**: Organize timed coding competitions to simulate real contest environments.
5. **Group Strategy Discussions**: Facilitate discussions on effective problem-solving strategies and approaches.
6. **Continuous Feedback**: Implement performance assessments and code reviews to enhance skills.

Outside Classroom Learning Experience

1. **Practice Assignments**: Assign challenging coding problems for independent practice from various platforms.
2. **Online Competitions**: Encourage participation in external coding contests and hackathons.
3. **Question Bank**: Provide a repository of advanced practice problems for self-assessment.
4. **Online Collaboration**: Create forums for students to discuss problems and share solutions.
5. **Self-Study Resources**: Recommend books and online courses focused on advanced algorithms and techniques.
6. **Collaborative Projects**: Organize group projects that involve developing coding solutions for complex problems.

Textbooks:

- "Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein

- "Data Structures and Algorithms Made Easy" by Narasimha Karumanchi

Online References:

1. **GeeksforGeeks**:
   - Offers articles on advanced data structures like self-balancing trees, segment trees, tries, and more[1].
   - Link to GeeksforGeeks
2. **Coursera**:
   - Various data structures and algorithms courses available online.
   - Examples include "Data Structures and Algorithms" from the University of California San Diego and "Algorithms, Part I" from Princeton University[3].
   - Link to Coursera
3. **Princeton University References**:
   - Provides a list of seminal papers and advanced resources.
   - Includes textbooks like "Algorithms, 4th Edition" by Robert Sedgewick and Kevin Wayne[4].

Lab Experiments

Problem Statement	Mapped COs
Object-Oriented Programming Concepts	
1. Design a Parking Lot System using OOP concepts (Classes, Objects, Inheritance, Polymorphism).	CO1
2. Implement a Student Management System with Classes and Objects.	CO1
3. Create a Banking System with Constructors and Destructors.	CO1

4. Implement Method Overloading and Overriding in a chosen language.	CO1
5. Demonstrate Multiple Inheritance with a practical example.	CO1
6. Design a Library Management System with OOP principles.	CO1
7. Use Virtual Functions to implement polymorphism.	CO1
Problem Statement	**Mapped COs**
8. Implement Abstract Classes and Interfaces for a Payment System.	CO1
9. Create a simple calculator with Operator Overloading.	CO1
10. Build a Polymorphic class hierarchy (e.g., Shapes) to showcase polymorphism.	CO1
Linked Lists, Stacks, and Queues	
11. Reverse a Linked List (Iterative and Recursive).	CO2
12. Detect a cycle in a Linked List using Floyd's Cycle-Finding Algorithm.	CO2
13. Implement basic operations on a Singly Linked List (Insertion, Deletion).	CO2
14. Implement and traverse a Doubly Linked List.	CO2
15. Implement Stack operations (Push, Pop, Top) using arrays or linked lists.	CO2
16. Implement Queue operations (Enqueue, Dequeue, Front) using arrays or linked lists.	CO2
17. Solve the Parentheses Matching problem using Stack.	CO2

18. Implement Sliding Window Maximum using Deque.	CO2
19. Check for balanced parentheses using Stack.	CO2
20. Design a Circular Queue using linked list or array.	CO2
Sorting & Searching	
21. Implement Bubble Sort and analyze its time complexity.	CO3
Problem Statement	**Mapped COs**
22. Implement Merge Sort to sort an array of integers.	CO3
23. Find the Kth largest element in an array using Quick Sort.	CO3
24. Perform Binary Search to find an element in a sorted array.	CO3
25. Implement Heap Sort to sort a list of elements.	CO3
26. Find the position to insert an element in a sorted array using Binary Search.	CO3
27. Implement a custom sort based on frequency of elements.	CO3
28. Compare sorting results using Insertion Sort and Bubble Sort.	CO3
Trees	
29. Perform Preorder, Inorder, and Postorder Traversal on a Binary Tree.	CO4
30. Find the Lowest Common Ancestor in a Binary Search Tree.	CO4

Problem Statement	**Mapped COs**
Trees	

31. Implement an algorithm to check if a Binary Tree is balanced.	CO4
32. Determine if two Binary Trees are identical.	CO4
33. Find the maximum path sum in a Binary Tree.	CO4
34. Convert a Binary Search Tree to a Greater Tree (where each node's value is replaced by the sum of all greater values).	CO4
Problem Statement	**Mapped COs**
35. Count the number of nodes in a complete Binary Tree.	CO4
36. Flatten a Binary Tree to a linked list using preorder traversal.	CO4
37. Serialize and deserialize a Binary Tree.	CO4
38. Find the diameter of a Binary Tree (the longest path between any two nodes).	CO4
39. Check if a Binary Tree is a subtree of another Binary Tree.	CO4
40. Find the level order traversal of a Binary Tree (Breadth-First Search).	CO4

Semester V

ANALYSIS AND DESIGN OFALGORITHMS

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Analysis and Design of Algorithms	**ENCS202**	4-0-0	4
Type of Course:	**Major**		
Pre-requisite(s), if any: - Data Structure			

Course Perspective The course provides a comprehensive introduction to the fundamental concepts of algorithm analysis and design, essential for various fields such as computer science, engineering, data science, and artificial intelligence. This course equips students with the tools to understand, analyze, and develop efficient algorithms for solving complex computational problems. By covering both theoretical foundations and practical applications, the course ensures a balanced approach to learning. The course is divided into 5 modules:

a) Introduction and Complexity Analysis

b) Divide and Conquer, Greedy Algorithms, and Dynamic Programming

c) Graph Algorithms

d) Advanced Algorithms and Techniques

e) Advanced Topics and Implementation Techniques

The Course Outcomes (COs)

COs	Statements
CO 1	**Understanding** fundamental algorithmic concepts and analyze their complexities.

CO 2	**Analyzing** and evaluate the performance of various algorithms.
CO 3	**Designing** efficient algorithms considering both time and space complexities.
CO 4	**Applying** algorithmic problem-solving strategies to a variety of computational problems.
CO 5	**Developing** skills to implement and optimize algorithms for real-world applications.

Course Outline:

Unit Number: 1	**Title:** Introduction and Complexity Analysis	**No. of hours: 10**
Content Summary: **Introduction to Algorithms**: Definition, importance, specification and role in problem-solving. **Algorithm Analysis**: RAM computational models, Time and space complexity, Asymptotic Notations, best, average, and worst-case analysis, Performance measurement of algorithms,rate of growth of algorithms **Recurrence Relations**: Solving recurrences using substitution, recursion tree, and mastertheorem.		
Unit Number: 2	**Title:** Divide and Conquer, GreedyAlgorithms, and Dynamic Programming	**No. of hours: 10**
Content Summary: **Divide and Conquer**: General method, Merge Sort, Quick Sort, Binary Search, Strassen's Matrix Multiplication, finding maximum and minimum.		
Greedy Algorithms: Concept and characteristics, Fractional Knapsack, Activity Selection,Huffman Coding. **Dynamic Programming**: General Method, Longest Common Subsequence, 0/1 Knapsackproblem, Matrix Chain Multiplication, Travelling salesman problem.		
Unit Number: 3	**Title:** Graph Algorithms	**No. of hours: 10**

Content Summary: **Graph Representation**: Adjacency matrix, adjacency list. **Graph Traversal Algorithms**: Depth First Search (DFS), Breadth First Search (BFS),Applications of graph (Topological sorting). **Shortest Path Algorithms**: Dijkstra's algorithm, Bellman-Ford algorithm, Floyd-Warshallalgorithm. **Minimum Spanning Tree Algorithms**: Kruskal's algorithm, Prim's algorithm.		
Unit Number: 4	**Title**: Advanced Algorithms and Techniques	**No. of hours: 10**
Content Summary: **Backtracking**: Concept, examples (N-Queens problem, Sum of subsets). **Branch and Bound**: Concept, examples (Traveling Salesman Problem, 0/1 KnapsackProblem). **String Matching Algorithms**: Naive algorithm, Rabin-Karp algorithm, String matching withfinite automata, Knuth-Morris-Pratt (KMP) algorithm. **Introduction to NP-Completeness:** The class P and NP, Polynomial time, NP-complete andNP-hard. Introduction to Approximation Algorithms and Randomized Algorithms		

Text Books

1. Introduction to Algorithms, 4TH Edition, Thomas H Cormen, Charles ELieserson, Ronald L Rivest and Clifford Stein, MIT Press/McGraw-Hill.

2. Fundamentals of Algorithms – E. Horowitz et al.

Additional Readings:

Online Learning Resources :

I) MIT OpenCourseWare - Introduction to Algorithms (6.006)

a. A comprehensive resource from MIT covering fundamental and advanced algorithms.

b. Link: MIT OpenCourseWare - Introduction to Algorithms

II) HackerRank - Algorithms Practice

a. Provides a platform to practice and compete in coding challenges relatedto algorithms.

b. Link: HackerRank - Algorithms Practice

III) LeetCode - Algorithm Problems

a. A platform offering a vast array of problems to practice algorithms anddata structures.

b. Link: LeetCode - Algorithm Problems

IV) TopCoder - Algorithm Tutorials

a. Detailed tutorials and practice problems on a wide range of algorithmictopics.

b. Link: TopCoder - Algorithm Tutorials

ANALYSIS AND DESIGN OF ALGORITHMS LAB

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Analysis and Design of Algorithms Lab	**ENCS256**	0-0-2	1
Type of Course:	Major		
Pre-requisite(s), if any: - Data Structure			

Defined Course Outcomes

COs	Statements
CO 1	**Analyzing** the time and space complexity of algorithms, demonstrating anunderstanding of asymptotic notations and performance metrics.
CO 2	**Implementing** and compare sorting algorithms, such as bubble sort and insertion sort, and apply the divide and conquer technique to algorithms like merge sort and quick sort
CO 3	**Solving** optimization problems using greedy and dynamic programming algorithms, such as the fractional knapsack problem and longest commonsubsequence
CO 4	**Developing** graph algorithms for traversal, shortest path, and minimum spanning tree, applying techniques like DFS, BFS, Dijkstra's, and Kruskal'salgorithms
CO 5	**Implementing** advanced algorithms for problems like N-Queens, traveling salesman, and string matching using backtracking, branch and bound, and pattern matching techniques

Proposed Lab Experiments

S.N	Lab Task	Mapped CO/COs
1	**Project 1: Case Study on Algorithm Efficiency** **Problem Statement:** Conduct a case study on the efficiency of different sortingalgorithms (e.g., Bubble Sort, Merge Sort, Quick Sort). Analyze and compare their time and space complexities in best, average, and worst-case scenarios. Presentyour findings in a detailed report with visual aids (graphs, charts).	CO1
2	**Project 2: Performance Measurement Tool** **Problem Statement:** Develop a tool in your preferred programming language to measure the performance of various algorithms. The tool should allow users to input an algorithm, run it with different input sizes, and display the time and space complexities. Implement a few sample algorithms (e.g., linear search, binary search)to demonstrate the tool's capabilities.	CO1
3	**Project 3: Resource Allocation System** **Problem Statement:** Develop a resource allocation system for a fictional company using greedy algorithms. Implement solutions for problems such as the Fractional Knapsack problem (for resource allocation), Activity Selection problem (for task scheduling), and Huffman Coding (for data compression). The system should allow users to input data and display the optimal allocation/selection in real-time.	CO2
4	**Project 4: Dynamic Programming Solver** **Problem Statement:** Build a web application that solves dynamic programming problems. Implement solutions for the Longest Common Subsequence, 0/1 Knapsack Problem, and Matrix Chain Multiplication. Allow users to input problem parameters and visualize the step-by-step solution process, including the dynamic programmingtable updates.	CO2
5	**Project 5: Smart City Traffic Management System** **Problem Statement:** Develop a smart traffic management system for a city to optimize traffic flow and reduce congestion. The system will use graph algorithms to model the city's road network and find the shortest paths, optimal routes, and manage traffic signals efficiently.	CO3
6	**Project Title 6: Intelligent Scheduling System**	CO4

	Problem Statement: Create an intelligent scheduling system for a university to optimize the allocation of classes, rooms, and faculty schedules. The system should use advanced algorithms like backtracking, branch and bound, and approximation algorithms to ensure efficient and conflict-free scheduling.	

INTRODUCTION TO COMPUTER ORGANIZATION & ARCHITECTURE

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Introduction to Computer Organization & Architecture	**Course Code**	**L-T-P**	**Credits**
	ENCA302	3-1-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Concepts of Digital Electronics			

Course Perspective: The course provides an in-depth understanding of the structural and operational principles of computer systems. It explores the intricate details of computer architecture, bridging the gap between high-level programming and hardware design. Students will learn about the functional units of a computer, instruction set architectures, processor design, memory hierarchy, and I/O systems. This knowledge is crucial for designing efficient, high-performance computing systems and understanding the trade-offs in hardware and software design. The course is divided into 4 units:

a) Introduction
b) Instruction Set Architecture (RISC-V)
c) The Processor
d) Memory hierarchy, Storage, and I/O

Course Outcomes (COs): On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the basics of instructions sets and their impact on processor design.
CO 2	**Demonstrating** an understanding of the design of the functional units of a digital computer system
CO 3	**Evaluating** cost performance and design trade-offs in designing and

	constructing a computer processor including memory.
CO 4	**Designing** a pipeline for consistent execution of instructions with minimum hazards
CO 5	**Manipulating** representations of numbers stored in digital computers using I/O devices and store them into memory

Course Outline:

Unit Number: 1	**Title: Introduction**	**No. of hours: 10**
Content : **Introduction to Computer Architecture:** Definitions and Concepts, Levels of abstraction, Von Neumann Architecture. **Functional Blocks of a Computer:** CPU, memory, input-output subsystems, control unit. **Instruction Set Architecture (ISA) of CPU:** Registers, instruction execution cycle, RTL (Register Transfer Language) interpretation of instructions, addressing modes, instruction set. **Types of Instruction Set Architectures:** Reduced Instruction Set Computer (RISC) and Complex Instruction Set Computer (CISC).		
Unit Number: 2	**Title: Memory hierarchy, Storage and I/O**	**No. of hours: 10**
Content : **Memory Hierarchy:** Types of memory: RAM (Random Access Memory), ROM (Read-Only Memory), Cache, and Secondary Storage, SRAM (Static RAM) vs. DRAM (Dynamic RAM), Locality of reference: spatial and temporal locality.		

Caching: Different indexing mechanisms: direct-mapped, set-associative, fully associative, Trade-offs related to block size, associativity, and cache size, Processor-cache interactions for read/write requests, Basic optimizations: write-through vs. write-back caches, Average memory access time (AMAT), Cache replacement policies: Least Recently Used (LRU), First-In-First-Out (FIFO).		
Unit Number: 3	**Title: The Processor**	**No. of hours: 10**
Content: **Building a Datapath:** Introduction, Logic Design Conventions, Building a Datapath: A Simple Implementation scheme, Overview of Pipelining: Pipelined Datapath and Control, Data Hazards: Forwarding versus Stalling, Control Hazards and their mitigation, Handling Exceptions, Parallelism via Instructions. **Processor Design:** Single cycle processor design, Multi-cycle processor design, Instruction pipelining: stages and performance considerations, Notion of Instruction Level Parallelism (ILP), Data and control hazards: identification and mitigation strategies.		
Unit Number: 4	**Title: Input/output Systems and Advanced Topics**	**No. of hours: 10**
Content Summary: **I/O Systems: I**/O Mapped vs. Memory-Mapped I/O, I/O Data Transfer Techniques: Programmed I/O, Interrupt-Driven I/O, Direct Memory Access (DMA) **Storage Technologies:** Introduction to Magnetic Disks: Tracks, Sectors, Disk Scheduling Algorithms, Flash Memory Technology: Structure and Performance Characteristics **Cache Memory:** Different Indexing Mechanisms: Direct-Mapped, Set-Associative, Fully Associative Caches, Trade-offs related to Block Size, Associativity, and Cache Size, Processor-Cache Interactions for Read/Write Requests, Basic Optimizations: Write-Through vs. Write-Back Caches, Average Memory Access Time (AMAT), Cache Replacement Policies: Least Recently Used (LRU), First-In-First-Out (FIFO).		

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key concepts in computer organization using PPTs and diagrams.
2. **Conceptual Understanding**: Cover topics like CPU architecture, memory hierarchy, and I/O systems.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on assembly language and instruction set architecture.
4. **Case Studies**: Analyze real-world computer architectures and their performance metrics.
5. **Group Work**: Collaborate on projects that involve designing and simulating computer systems.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding of organizational concepts.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign projects that require application of concepts in computer organization and architecture.
2. **Lab Projects**: Facilitate hands-on tasks involving hardware simulations and assembly programming.
3. **Question Bank**: Provide practice problems and resources for self-assessment on architecture topics.
4. **Online Forums**: Create platforms for discussing challenges and sharing solutions related to computer architecture.
5. **Self-Study for Case Studies**: Encourage independent research on current trends in computer organization.
6. **Collaborative Projects**: Organize group projects focused on building and optimizing computer architectures.

Text Books:

1. David A. Patterson and John L. Hennessy ,"Computer Organization and Design: The Hardware/Software Interface",5th Edition, Elsevier
2. Mano M. Morris, "Computer System Architecture", Pearson.

Reference Books:

1. CarlHamache, "Computer Organization and Embedded Systems", 6th Edition, McGraw Hill Higher Education
2. "Computer Architecture and Organization", 3rd Edition by John P. Hayes, WCB/McGraw-Hill
3. William Stallings "Computer Organization and Architecture: Designing for Performance", 10th Edition, Pearson Education

Additional Readings:

Online Learning References

a) **MIT OpenCourseWare - Computer System Engineering**

a. **Link:** MIT OCW

b. **Description:** This course provides a deep dive into computer system architecture, exploring processor design, memory systems, and parallel processing.

b) **GeeksforGeeks - Computer Organization and Architecture**

a. **Link:** GeeksforGeeks

b. **Description:** GeeksforGeeks provides detailed tutorials on various topics in computer organization and architecture, such as instruction sets, pipelining, and memory hierarchy.

c) **NPTEL - Computer Architecture**

a. **Link:** NPTEL

b. **Description:** This course from NPTEL covers the principles of computer architecture, including instruction sets, CPU design, and memory systems, with a focus on practical applications.

NATURAL LANGUAGE PROCESSING

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
NATURAL LANGUAGE PROCESSING	**ENSP302**	4-0-0	4
Type of Course:	Minor (Department Elective II)		
Pre-requisite(s), if any: Strong programming skills, particularly in Python.			

Course Perspective: This course offers a comprehensive introduction to the principles and techniques of Natural Language Processing (NLP). It bridges the gap between theoretical linguistics and practical implementation using machine learning and deep learning algorithms. NLP is pivotal in developing applications that can interpret and respond to human language, impacting areas such as artificial intelligence, data analysis, and information retrieval. The course emphasizes hands-on learning through real-world applications, ensuring that students gain practical skills in processing and analyzing natural language. The course is divided into 4 modules:

- Foundations of Natural Language Processing
- Machine & Deep Learning in Natural Language Processing
- Advanced Text and Speech Processing
- Ethics, Fairness, and Practical Applications in NLP

The Course Outcomes (COs): On completion of the course the participants willbe:

COs	Statements
CO 1	**Grasping** NLP fundamentals, resolve linguistic ambiguities, and employML/DL in NLP for both text and speech.

CO 2	**Applying** techniques for processing and analyzing natural language to develop intelligent interpretation systems.
CO 3	**Designing** and assess advanced NLP models for essential tasks, utilizing transformers, GRUs, and more.
CO 4	**Inventing** with NLP and speech recognition for real-world applications, incorporating cutting-edge concepts.
CO 5	**Evaluating** NLP and speech technologies' fairness and ethics, striving for inclusive and meaningful solutions.

Course Outline:

Unit Number: 1	**Title: Foundations of Natural LanguageProcessing**	**No. of hours: 10**
Content Summary: **Overview of NLP and its applications**: Basic text processing and analysis, Introduction to Python for NLP, Language modeling: N-grams, smoothing techniques, Morphology and Parts of Speech (PoS) Tagging, Syntax: Probabilistic Context-Free Grammars (PCFGs), Dependency Parsing, Lexical semantics and Word Sense Disambiguation, Distributional semantics: Vector space models, Word2Vec, Introduction to GloVe for Word Representation, Topic Models: Latent Dirichlet Allocation (LDA).		
Unit Number: 2	**Title: Machine & Deep Learning inNatural Language Processing**	**No. of hours: 12**
Content Summary: **Introduction to Neural Networks** for NLP, Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, Introduction to Gated Recurrent Units (GRUs)and their comparison with LSTMs, Deep Learning Models: Basic to Advanced Architectures **Neural & Pre-Trained Language Models:** Introduction and Applications, Advanced Pre-training Techniques for Language Models, Generative models & Beyond: Few-Shot Learning, Prompt Learning, and Applications, Recursive Neural Networks.		

Unit Number: 3	Title: Advanced Text and Speech Processing	No. of hours: 10
Content Summary: **Advanced text classification techniques** using Naïve Bayes, Logistic Regression, and SVMs, alongside applications in sentiment analysis, opinion mining, and text summarization. It includes N-gram feature applications and enhancements through CNNs, challenges in multilingual NLP, and advanced entity linking. The section also covers the basics of speech recognition and synthesis, speech-to-text and text-to-speech technologies, and speaker recognition and verification.		
Unit Number: 4	**Title: Ethics, Fairness, and Practical Applications in NLP**	**No. of hours: 8**
Content Summary: **Fairness and ethical considerations in NLP**, with a focus on developing real-world NLP solutions through application-oriented projects. Topics include spelling correction, language modeling for robust applications, and ethical speech recognition system design. The syllabus also explores multilingual speech processing, applications of large language models in creative text generation, summarization, and code generation. Additionally, practical applications of speech recognition in assistive technologies, automated customer service, and real-time transcription services are covered.		

Text Books:

1. Daniel Jurafsky and James H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*, Pearson Education India.
2. Steven Bird, Ewan Klein, and Edward Loper, *Natural Language Processingwith Python*, O'Reilly Media.

References

1. "Speech and Language Processing" by Dan Jurafsky and James H. Martin
2. "Natural Language Processing in Action: Understanding, Analyzing, and Generating Text with Python" by Hobson Lane, Cole Howard, and Hannes Hapke
3. "Neural Network Methods for Natural Language Processing" by Yoav Goldberg
4. "Introduction to Information Retrieval" by Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze
5. "Practical Natural Language Processing: A Comprehensive Guide to Building

Real-World NLP Systems" by Sowmya Vajjala, Bodhisattwa Majumder, Anuj Gupta, and Harshit Surana

Additional Readings:

Online Learning Resources:

1. **Stanford University - Natural Language Processing with DeepLearning (CS224N)**

- Offered by Stanford University, this course covers the foundational concepts of NLP along with the application of deep learning models inNLP tasks. The lectures are available for free.
- Link: Stanford CS224N

2. **NLTK Book - Natural Language Processing with Python**

- An excellent resource for beginners, this book provides a practical introduction to NLP with Python, using the Natural Language Toolkit(NLTK). It's available for free online.
- Link: NLTK Book

3. **fast.ai - Practical Deep Learning for Coders**

- Though not exclusively focused on NLP, this course includes several lessons on applying deep learning to natural language processing. It ispractical, with a focus on coding and real-world applications.
- Link: fast.ai Course

Open-Source Society University (OSSU)

I) OSSU Computer Science

a. OSSU provides an open-source curriculum for learning computer science. While it covers a broad range of topics in computer science, it includes resources for learning about artificial intelligence and machinelearning, which are relevant to students interested in NLP.

b. Link: OSSU Computer Science

NATURAL LANGUAGE PROCESSING LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
NATURAL LANGUAGE PROCESSING LAB	**ENSP352**	0-0-2	1
Type of Course:	Minor (Department Elective II)		
Pre-requisite(s), if any: Strong programming skills, particularly in Python.			

Proposed Lab

ExperimentsDefined Course Outcomes

COs	
CO 1	**Demonstrating** proficiency in implementing and utilizing basic and advanced natural language processing (NLP) techniques for text analysis and processing.
CO 2	**Developing** fine-tune, and evaluate machine learning and deep learning models for various NLP tasks, including text classification, named entity recognition, and sentiment analysis.
CO 3	**Implementing** and assess advanced text and speech processing models, focusing on performance improvement and practical applications in real-world scenarios.
CO 4	**Analyzing** and address ethical considerations, fairness, and privacy implications in NLP applications, ensuring responsible and unbiased technology deployment.

List of Experiments

Project Title	Project Statement	CO Mapped
Text Processing and Analysis Tool	Develop a Python-based tool for basic text processing and analysis. Implement functionalities such as tokenization, stemming, lemmatization,and Parts of Speech (PoS) tagging. Include language modeling using N-grams and smoothing techniques.	CO1
Word Representation and Topic Modeling	Create a Python application to perform word representation using Word2Vec and GloVe. Implement a topic modeling system using Latent Dirichlet Allocation (LDA) to analyze and categorize a given corpus of text data.	CO1
Sentiment Analysis using Recurrent Neural Networks (RNNs)	Build a sentiment analysis application using Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks. Train the model on a labeled dataset and evaluate its performance in classifying sentiments.	CO2
Language Model with Gated Recurrent Units (GRUs)	Develop a language model using Gated Recurrent Units (GRUs). Compare its performance with LSTM-based models on the same dataset. Implement applications for text generation based on the trained language model.	CO2
Advanced Text Classification and Sentiment Analysis	Implement text classification techniques using Naïve Bayes, Logistic Regression, and SVMs. Apply these techniques to sentiment analysis, opinion mining, and text summarization tasks. Enhance feature extraction using N-grams and CNNs.	CO3

Project Title	Project Statement	CO Mapped
Speech Recognition and Synthesis System	Develop a basic speech recognition and synthesis system. Implement speech-to-text and text-to-speech functionalities using available libraries. Extend the system to include speaker recognition and verification.	CO3
Ethical NLP System for Fair Language Modeling	Design an NLP system focusing on ethical considerations and fairness. Implement language modeling for robust applications like spelling correction. Ensure the system avoids biases and treats all user inputs fairly.	CO4
Real-World NLP Application: Automated Customer Service	Create an automated customer service application using advanced language models. Implement functionalities such as real-time transcription services and multilingual speech processing. Ensure ethical and fair use of NLP technologies.	CO4

LIFE SKILLS FOR PROFESSIONALS-III

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Life Skills for Professionals -III	**AEC013**	3-0-0	3
Type of Course:	AEC		
Pre-requisite(s), if any:			

Course Perspective. The course is designed to equip students with essential skills for both professional and personal development. This course emphasizes the importance of data interpretation, logical reasoning, stress management, and employability skills, ensuring students are well-prepared to tackle real-world challenges. Through a combination of theoretical knowledge and practical exercises, students will enhance their analytical abilities, problem-solving skills, and emotional intelligence. The course is structured into five comprehensive modules, each targeting a specific skill set vital for professional success and personal well-being:

a) Data Interpretation

b) Logical Reasoning

c) Logical & Non-verbal Reasoning

d) Understanding Stress

e) Employability Skills

The Course Outcomes (COs): On completion of the course the participants will be:

The Course Outcomes (COs): On completion of the course the participants will be able to:

COs	Statements
CO 1	**Understanding** their critical thinking skills and become adept at analyzing and evaluating information, identifying problems, generating innovative solutions, and making informed decisions.
CO 2	**Applying** digital literacy skills necessary for the modern workplace and become proficient in using online platforms relevant to their field.
CO 3	**Evaluating** Contribute positively, respect different perspectives, resolve conflicts, and achieve shared goals.
CO 4	**Improving** skills related to career planning, job search strategies, and personal branding.
CO 5	**Creating** leadership skills and to motivate and inspire others, manage projects effectively, and demonstrate a proactive and responsible approach to their spoken language.

Course Outline:

Unit Number: 1	**Title: Data interpretation**	**No. of hours: 6**
Content Summary: Table chart, Line graph, Bar graph, Pie chart		
Unit Number: 2	**Title: Logical Reasoning**	**No. of hours: 6**
Content Summary: Coding & Decoding, Sitting arrangement, Calendar, Clock, Direction Sense, Blood relation, Syllogism.		
Unit Number: 3	**Title: Logical & Non-verbal reasoning**	**No. of hours: 6**

Content Summary: Series, Puzzle Text, Statement & Arguments, Cube & Dice, Non-verbal Reasoning		
Unit Number: 4	**Title: Understanding Stress**	**No. of hours: 6**
Content Summary: Introduction to Stress (i) Introduction to stress: Meaning, Definition, Eustress, Distress, (ii) Types of stress: Acute stress, Episodic Acute stress and chronic stress, signs and Symptoms Sources of stress (i) Psychological, Social, Environmental (ii) Academic, Family and Work stress Impact of stress		
Unit Number: 5	**Title: Employability skills**	**No. of hours: 6**
Content Summary: Identifying job openings, enhancing interpersonal skills, including teamwork, Applying for a job, Preparing Cover letters, preparing a CV/Resume and Effective Profiling, Group Discussions, Preparing for and Facing a Job Interview, Mock Interview, Feed Back – Improvement		

References

1. Aggarwal, R. S. (2014). Quantitative aptitude (Revised edition).
2. Gladwell, M. (2021). Talking to strangers.
3. Scott, S. (2004). Fierce conversations.

Introduction to Computer Networks

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Introduction to Computer Networks	**Course Code**	**L-T-P**	**Credits**
	ENCA304	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Basics of Communication Engineering and Data structures			

Course Perspective: The course provides a comprehensive introduction to the fundamental concepts and principles of computer networking. Students will gain an in-depth understanding of how data communication works, the different types of network topologies, and the protocols that govern data exchange over networks. The course covers key aspects such as network hardware and software components, network administration, and protocol selection for various communication services. By exploring topics such as the OSI model, error detection and correction, IP addressing, routing protocols, and network applications, students will develop the skills necessary to design, implement, and manage computer networks effectively. The course is divided into 4 modules:

a) Evolution of Computer Networking
b) Data Link Layer Design
c) Introduction to Network Layer and Transport Services
d) Application Layer

The Course Outcomes (COs). On completion of the course the participants will be able to:

COs	Statements
CO 1	**Understanding** the fundamental concepts and principles of computer networks.
CO 2	**Applying** knowledge of network hardware and software components.
CO 3	**Developing** skills in network administration and management.
CO 4	**Identifying** appropriate protocol for desired communication service.

Course Outline:

Unit Number: 1	**Title: Evolution of Computer Networking**	**No. of hours: 10**
Content Summary: **Data communication Components**: Representation of data and its flow Networks, Various Connection Topology, Protocols and Standards, OSI model; packet switching; circuit switching;		
Unit Number: 2	**Title: Data Link Layer**	**No. of hours: 10**
Content Summary: **Data Link Layer and Medium Access Sub Layer**: Error Detection and Error Correction Fundamentals, Block coding, Hamming Distance, CRC; Flow Control and Error control protocols - Stop and Wait, Go back – N ARQ,		
Unit Number: 3	**Title: Introduction to Network Layer and Transport Services**	**No. of hours: 10**

Content Summary: **Network Layer:** Switching, Logical addressing – IPV4, IPV6; **Address mapping** – ARP, RARP, BOOTP and DHCP–Delivery, Forwarding and Unicast Routing protocols. **Transport Layer:** Process to Process Communication, User Datagram Protocol (UDP), Transmission Control Protocol (TCP)		
Unit Number: 4	**Title: Application Layer**	**No. of hours: 10**
Content Summary: **Application Layer:** Domain Name Space (DNS), DDNS, TELNET, EMAIL, File Transfer Protocol (FTP), WWW, HTTP, SNMP, Bluetooth, Firewalls		

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key concepts in computer networks using PPTs and network diagrams.
2. **Conceptual Understanding**: Cover topics like network protocols, topologies, and architecture models (OSI, TCP/IP).
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on troubleshooting network issues and configuring devices.
4. **Case Studies**: Analyze real-world network designs and their performance implications.
5. **Group Work**: Collaborate on projects that involve designing and simulating network setups.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding of networking concepts.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign projects that require practical application of networking principles.
2. **Lab Projects**: Facilitate hands-on tasks involving network configuration and management tools.

3. **Question Bank**: Provide practice problems and resources for self-assessment on networking topics.
4. **Online Forums**: Create platforms for discussing networking challenges and sharing solutions.
5. **Self-Study for Case Studies**: Encourage independent research on current trends and technologies in computer networking.
6. **Collaborative Projects**: Organize group projects focused on developing efficient network solutions for real-world scenarios.

Textbooks:

T1: " Data Communication and Networking", 5th Edition, Behrouz A. Forouzan, McGraw-Hill, 2012.

T2: "Computer Networks", Andrew S. Tanenbaum and David J. Wetherall, Pearson, 5th Edition, 2010.

T3: "Data and Computer Communication", 8th Edition, William Stallings, Pearson Prentice Hall Indi

Reference Books

R1: "Computer Networking A Top-Down Approach". 5th Edition, James F. Kurose-Keith W. Ross (Pearson).

R3. "Computer Networks – Protocols, Standards and Interfaces". 2nd Edition – Uyless Black (Prentice Hall of India Pvt. Ltd.

Additional Readings:

Online Learning References

I) MIT OpenCourseWare - Computer Networks

a. **Link:** MIT OCW

b. **Description:** This course provides a thorough exploration of computer networks, focusing on network design, protocol layers, and network management.

II) GeeksforGeeks - Computer Networks

a. **Link:** GeeksforGeeks

b. **Description:** GeeksforGeeks provides detailed tutorials on various aspects of computer networks, such as the OSI model, data link layer, network layer, and transport layer protocols.

III) NPTEL - Computer Networks

a. **Link:** NPTEL

b. **Description:** This course from NPTEL provides a comprehensive overview of computer networking, including topics like error detection, IP addressing, and routing protocols.

COMPUTER NETWORKS LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Computer Networks Lab	**ENCS352**	0-0-2	1
Type of Course:	Major		
Pre-requisite(s), if any:			

Proposed Lab ExperimentsDefined Course Outcomes

COs	
CO 1	**Applying** fundamental networking concepts and techniques to develop and analyze network topologies, protocols, and error detection mechanisms.
CO 2	**Designing** and implement network protocols and architectures for efficient data communication and management in various environments.
CO 3	**Utilizing** advanced networking techniques to implement, monitor, and optimize communication systems for real-time and multimedia applications.
CO 4	**Integrating** IoT devices and develop smart systems using networking principles for automation and efficient data management.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Design and simulate a simple computer network using various connection topologies (bus, star, ring, mesh). Compare the advantages and disadvantages of each topology in terms of data flow and network efficiency.	CO 1
2	Create a network simulation to demonstrate packet switching and circuit switching. Compare the performance and efficiency of both methods by simulating a series of data transmission scenarios.	CO 1
3	Develop a network simulator to analyze packet delay, loss, and end-to-end throughput. Implement various routing algorithms and measure their impact on network performance under different traffic conditions.	CO1
4	Develop a simulation to demonstrate multiple access protocols such as Pure ALOHA, Slotted ALOHA, CSMA/CD, and CSMA/CA. Analyze the performance of each protocol in handling network collisions and maximizing data transmission efficiency	CO2
5	Implement a sliding window protocol with piggybacking for efficient data transmission and error control. Simulate data transfer between two nodes and visualize the window movements and acknowledgments.	CO2
6	Create a simulation to demonstrate logical addressing using IPv4 and IPv6. Implement address mapping techniques such as ARP, RARP, BOOTP, and DHCP to show how devices acquire and resolve network addresses.	CO3
7	Implement a transport layer simulation to demonstrate process-to-process communication using UDP, TCP, and SCTP. Compare the protocols in terms of connection establishment, data transmission, and congestion control.	CO3

8	Implement a DNS and DDNS simulation to demonstrate domain name resolution and dynamic updates. Create a simple client-server application that queries and updates the DNS records.	CO4
9	Create a web server simulation to demonstrate the workings of HTTP and WWW. Implement basic HTTP request and response handling, and simulate a simple web browsing session.	CO4

INTRODUCTION OF NEURAL NETWORK AND DEEP LEARNING

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Introduction to Neural Network and Deep learning	**Course Code**	**L-T-P**	**Credits**
	ENCS306	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Fundamentals of AI and Machine Learning			

Course Perspective: This course provides a comprehensive introduction to the fundamental concepts of neural networks and deep learning, which are crucial for the development of intelligent systems and advanced data analysis. Students will explore core topics including the basic structure and function of neural networks, feed forward neural networks, deep learning techniques, and probabilistic neural networks. The course emphasizes both theoretical understanding and practical implementation, preparing students to tackle real-world problems using advanced neural network models. The course is divided into 4 modules:

a) Introduction to Neural Networks and Linear Models

b) Supervised and Unsupervised Neural Networks

c) Deep learning and Regularization for Deep Learning

d) Optimization for Train Deep Models

The Course Outcomes (COs): On completion of the course the participants will be able to:

COs	**Statements**
CO 1	**Understanding** key concepts of neural networks and deep learning, including ANNs and their biological equivalents.
CO 2	**Applying** the Learning Networks in modeling real world systems
CO 3	**Analyzing** suitable architectures and hyperparameters for various neural networks to optimize tasks like image classification and NLP
CO 4	**Implementing** efficient algorithm for Deep Models and apply optimization strategies for large scale applications

Course Outline:

Unit Number: 1	**Title: Introduction to Neural Networks**	**No. of hours: 10**
Content: Human Brain, Model of an artificial Neuron, Basic concepts of Neural Networks, Fundamentals of Biological Neural Network and Artificial Neural Network, Evolution and characteristics of Neural Networks, Learning Methods – supervised, unsupervised and reinforcement, Taxonomy of Neural Network Architectures.		
Unit Number: 2	**Title: Supervised and Unsupervised Neural Networks**	**No. of hours: 10**
Content: Supervised learning: - Supervised Learning Networks, Perceptron Networks, Adaptive Linear Neuron, Back-propagation Network. Associative Memory Networks. Training Algorithms for pattern association. Unsupervised learning: - Introduction, Fixed Weight Competitive Nets, Maxnet, Hamming Network, Kohonen Self-Organizing Feature Maps, Learning Vector Quantization, Counter Propagation Networks, Adaptive Resonance		

Theory Networks.		
Unit Number: 3	**Title: Deep learning and Regularizationfor Deep Learning**	**No. of hours: 10**
Content: **I**ntroduction to Deep Learning, Historical Trends in Deep learning, Deep Feed - forward networks, Gradient-Based learning, Hidden Units, Architecture Design, Back- Propagation and Other Differentiation Algorithms **Regularization for Deep Learning:** Parameter norm Penalties, Norm Penalties as Constrained Optimization, Regularization and Under-Constrained Problems, Dataset Augmentation, Noise Robustness, Semi-Supervised learning, Multi-task learning, Early Stopping, Parameter Typing and Parameter Sharing, Sparse Representations, Bagging and other Ensemble Methods		
Unit Number: 4	**Title: Optimization for Train Deep Models**	**No. of hours: 10**
Content: Challenges in Neural Network Optimization, Basic Algorithms, Parameter Initialization Strategies, Algorithms with Adaptive Learning Rates, Approximate Second Order Methods, Optimization Strategies and Meta-Algorithms. **Applications:** Large-Scale Deep Learning, Computer Vision, Speech Recognition, Natural Language Processing recent trends and research in deep learning.		

Textbooks:

T1: Haykin S., Neural Network and Machine Learning, Prentice Hall Pearson (2009),3rd ed.

T2: Goodfellow L., Bengio Y. and Courville A., Deep Learning, MIT Press (2016).

References

I) "Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville

II) "Neural Networks and Deep Learning: A Textbook" by Charu C. Aggarwal

III) "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron

IV) "Deep Learning for Computer Vision" by Rajalingappaa Shanmugamani

Online Learning References

1. Deep Learning by Ian Goodfellow, Yoshua Bengio, and Aaron Courville

- Deep Learning Book

2. Neural Networks and Deep Learning by Michael Nielsen

- Neural Networks and Deep Learning

3. CS231n: Convolutional Neural Networks for Visual Recognition by Stanford University

- CS231n Course

4. Deep Learning Specialization by Andrew Ng on Coursera

- Deep Learning Specialization

5. Introduction to Deep Learning by MIT OpenCourseWare

- MIT OpenCourseWare

INTRODUCTION TO NEURAL NETWORKS & DEEP LEARNING LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Introduction to Neural Network and Deep learning Lab	**Course Code**	**L-T-P**	**Credits**
	ENCS354	0-0-2	1
Type of Course:	Major		
Pre-requisite(s), if any:			

Defined Course Outcomes

COs	
CO 1	**Demonstrating** the ability to implement and analyze basic artificial neural network models and classical machine learning algorithms.
CO 2	**Developing** and evaluate advanced neural network architectures and unsupervised learning models for complex data analysis.
CO 3	**Applying** deep learning techniques to real-world applications, including image and speech recognition, and optimize model performance.
CO 4	**Implementing** and optimize neural network models using advanced techniques such as adaptive learning rates, regularization, and meta-optimization for enhanced performance.

Proposed Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
P1	**Project 1: Neuron Model Simulation** **Problem Statement:** Develop a simulation of an artificial neuron using a programming language like Python. Implement the basic concepts of neural networks, including the neuron model, activation functions, and learning methods. Compare the behavior of artificial neurons with biological neural networks through visualizations and data analysis.	CO 1
P2	**Project 2: Supervised Learning Network Implementation** **Problem Statement:** Implement a supervised learning neural network using Python and libraries such as TensorFlow or PyTorch. Train the network using the Perceptron, Adaptive Linear Neuron, and Back-propagation algorithms on a dataset (e.g., MNIST for digit recognition). Evaluate the performance of each algorithm and present your results.	CO 2
P3	**Project 3: Unsupervised Learning Network Implementation** **Problem Statement:** Implement an unsupervised learning neural network using Python and libraries such as TensorFlow or PyTorch. Train the network using algorithms like Kohonen Self-Organizing Feature Maps and Learning Vector Quantization on a dataset (e.g., Iris dataset for clustering). Evaluate the network's ability to learn patterns without labeled data and present your results..	CO 3
P4	**Project 4: Deep Feed-Forward Network Development** **Problem Statement:** Develop a deep feed-forward neural network using Python and TensorFlow or PyTorch. Train the network on a dataset (e.g., CIFAR-10 for image classification) and implement various regularization techniques such as parameter norm	CO 4

	penalties, dataset augmentation, and early stopping. Analyze the impact of these techniques on the model's performance and generalization.	
P5	**Project 5: Optimization Algorithm Comparison** **Problem Statement:** Implement and compare various optimization algorithms, including basic algorithms, adaptive learning rate algorithms, and approximate second-order methods. Train a deep learning model on a dataset (e.g., ImageNet for large-scale image recognition) using these algorithms. Evaluate and present the effectiveness of each optimization strategy in terms of convergence speed and model accuracy.	CO4
P6	**Project 8: Real-World Application Development** **Problem Statement:** Develop a deep learning application for a real-world problem in computer vision, speech recognition, or natural language processing. Use advanced optimization strategies and meta-algorithms to enhance the model's performance. Document the challenges faced during model training and how optimization techniques helped overcome these challenges. Present a comprehensive report and demonstration of your application.	CO4

Online Learning References

1. **Deep Learning Specialization by Andrew Ng on Coursera**

- **Deep Learning Specialization**

2. **Neural Networks and Deep Learning by Michael Nielsen**

- **Neural Networks and Deep Learning**

3. **CS231n: Convolutional Neural Networks for Visual Recognition byStanford University**

- **CS231n Course**

4. **Deep Learning by Ian Goodfellow, Yoshua Bengio, and AaronCourville**

- **Deep Learning Book**

5. MIT OpenCourseWare - Introduction to Deep Learning

- **MIT OpenCourseWare**

CAREER READINESS BOOT CAMP

Program Name:	**Bachelor in Computer Applications (BCA)**		
Applied Programming and Problem-Solving Skills for Campus Interviews	**Course Code**	**L-T-P**	**Credits**
		---	2
Type of Course:	MOOC-II		
Pre-requisite(s), if any: NA			

Objective: The course "Applied Programming and Problem-Solving Skills for Campus Interviews" is designed to equip students with the essential technical and interpersonal skills required to excel in campus interviews and recruitment processes. By focusing on key areas such as Data Structures and Algorithms, Object-Oriented Programming, Java, and Database Management Systems, the course fosters problem-solving abilities critical for software development and technical roles. In addition to strengthening technical expertise, the course also prepares students for aptitude exams like AMCAT, enhances their soft skills, and simulates real-world interview scenarios through mock interviews. With certifications from Infosys Springboard, this comprehensive program ensures students are well-prepared for the demands of the modern job market.

All students are required to pass the individual components to receive the final marks out of 100 and earn 2 credits for the Practical Training Module in their course structure. Students must obtain specific **free certifications** from Infosys Springboard (https://infytq.onwingspan.com/web/en/page/home).

Course Outcomes (CO)

COs	Statements
CO1	**Understanding** and apply fundamental and advanced concepts in data structures, algorithms, object-oriented programming, and database

	management systems to solve complex problems efficiently
CO2	**Developing** and enhance programming skills using Java, demonstrating the ability to write, debug, and optimize code for real-world applications
CO3	**Analyzing** and solve aptitude and soft skills assessments, demonstrating proficiency in logical reasoning, quantitative ability, and effective communication
CO4	**Demonstrating** readiness for campus placements through successful participation in mock interviews, AMCAT assessments, and obtaining certifications from Infosys Springboard, showcasing industry-relevant skills and knowledge

Detailed Plan

Module	Link	Hrs
Data Structures and Algorithms - Part 1	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01254096991326208010 65_shared/overview	30
Data Structures and Algorithms - Part 2	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_0127667384693882883448_shared/overview	38
Object Oriented Programming	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_0125409722749255681063_shared/overview	46
Programming using Java	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01288046454761881634 7_shared/overview	113
Database management Systems (part I)	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_0127580666728202245 6_shared/overview	64

Database management Systems (Part II)	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_0127673005629194241_shared/overview	40
Aptitude Exam	Online Sessions to be conducted by KRMU	NA
AMCAT (Aspiring Minds Computer Adaptive Test)	To be organized by KRMU through External agency	NA
Softs Skills	Online Sessions to be conducted by KRMU	NA
MOCK Interview	To be organized by KRMU	

Important Notes

1. **Training on Soft Skills and Aptitude:** KRMU will organize online training sessions on Soft Skills and Aptitude components. Assessments will be conducted by the CDC.

2. **Certification Submission:** Full marks for certifications will only be awarded upon submission of the final certificate.

3. **AMCAT Exam Requirement:** Students are required to appear for the AMCAT exams. The AMCAT is a widely recognized employability assessment used by many companies to evaluate candidates' skills and job readiness. For the syllabus, please explore https://www.myamcat.com.

4. **MOCK Interviews:** These will focus on your preparation for placements, testing your coding skills, and preparation for technical and HR interview rounds.

5. Hard copies of the certificates for the specified certifications must be submitted to the Dean's office by the designated deadline. Certificates submitted after the deadline will not be accepted.

Learning Experiences

Classroom Learning Experience

1. **Interactive Workshops**: Engage in hands-on workshops focused on essential career skills.

2. **Problem-Based Assignments**: Tackle real-world challenges to enhance job readiness and critical thinking.

3. **Mock Interviews**: Participate in simulated interviews to practice communication and presentation skills.

4. **Collaborative Group Work**: Work in teams on projects that mimic workplace scenarios and dynamics.

5. **Continuous Feedback**: Receive regular assessments and constructive feedback from instructors.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle for flexible learning.

2. **Question Bank**: Utilize a question bank and resources for job application preparation.

Networking Opportunities: Participate in events to connect with industry professionals and expand your network.

(DEPARTMENT ELECTIVE -I)

IMAGE PROCESSING & COMPUTER VISION

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Image Processing & Computer Vision**	**Course Code**	**L-T-P**	**Credits**
	ENSP304	4-0-0	4
Type of Course:	Minor (Department Elective I)		
Pre-requisite(s), if any: (1) Linear Algebra and (2) programming in python			

Course Perspective. This course introduces students to the essential concepts and techniques of image processing and computer vision. It bridges theoretical knowledge with practical applications, enabling students to understand and implement various algorithms for image enhancement, restoration, segmentation, and object recognition. The course is designed to equip students with the skills required to tackle real-world challenges in fields such as robotics, medical imaging, surveillance, and multimedia applications.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Remembering** key concepts, definitions, and algorithms in image processing and computer vision.
CO 2	**Understanding** the principles and applications of essential techniques like edge detection and feature extraction.

CO 3	**Applying** Use image processing methods to enhance digital images and analyze the effects.
CO 4	**Evaluating** the effectiveness of various computer vision models in different contexts
CO 5	**Designing** and implement projects that integrate multiple image processing algorithms to solve complex problems.

Course Outline:

Unit Number: 1	**Title: Title: Introduction to Basic Concepts of Image Formation**	**No. of hours: 10**

Content Summary:

Fundamentals and Applications of Image Processing:

- Overview of Image Processing and its applications.
- Components of Image Processing Systems.

Image Sensing and Acquisition:

- Image sensing techniques.
- Image acquisition methods.

Sampling and Quantization:

- Concept of sampling and quantization.
- Neighbors of pixel adjacency and connectivity.
- Regions and boundaries.
- Distance measures.

Image Enhancement:

- Frequency and Spatial Domain techniques.
- Contrast Stretching.
- Histogram Equalization.
- Low pass and High pass filtering.

Unit Number: 2	**Title: Image Restoration and coloring**	**No. of hours: 12**

Content Summary:

Image Restoration:

- Model of the Image Degradation/Restoration Process.
- Noise Models.
- Restoration in the presence of noise using spatial filtering.
- Periodic Noise Reduction using frequency domain filtering.
- Linear Position Invariant Degradations.
- Estimation of Degradation Function.
- Inverse Filtering.
- Wiener Filtering.
- Constrained Least Square Filtering.
- Geometric Mean Filter.
- Geometric Transformations.

Color Image Processing:

- Color models and transformations.
- Image Segmentation using color.
- Texture Descriptors.
- Color Features.
- Edge/Boundary detection.
- Object Boundary and Shape Representations.
- Interest or Corner Point Detectors.
- Speeded-Up Robust Features (SURF).
- Saliency detection.

Unit Number: 3	Title: Image Compression and Segmentation	No. of hours: 10

Content Summary:

Image Compression:

- Data Redundancies.
- Image Compression models.
- Elements of Information Theory.
- Lossless and Lossy Compression.

- Huffman Coding.
- Shanon-Fano Coding.
- Arithmetic Coding.
- Golomb Coding.
- LZW Coding.
- Run Length Coding.
- Lossless Predictive Coding.
- Bit Plane Coding.
- Image compression standards.

Image Segmentation and Morphological Image Processing:

- Discontinuity-based segmentation.
- Similarity-based segmentation.
- Edge linking and boundary detection.
- Thresholding.
- Region-based Segmentation.
- Introduction to Morphology.
- Dilation and Erosion.
- Basic Morphological Algorithms.

Unit Number: 4	Title: Object Representation and Computer Vision Techniques	No. of hours: 8

Content Summary:

Representation and Description:

- Introduction to Morphology.
- Basic Morphological Algorithms.
- Representation Techniques.
- Boundary Descriptors.
- Regional Descriptors.
- Chain Code.
- Structural Methods.

Computer Vision Techniques:

- Review of Computer Vision Applications.

- Artificial Neural Networks for Pattern Classification.
- Convolutional Neural Networks (CNNs).
- Machine Learning Algorithms and their Applications

Learning Experiences:

- Lecture PPTs will offer clear, structured explanations to introduce and reinforce key concepts.
- Problem-based assignments will encourage deep thinking and application of theory to real-world scenarios.
- Project-based lab assignments will provide hands-on experience, making learning practical and engaging.
- Question banks will help students thoroughly review and self-assess their understanding of the entire syllabus.
- Model question papers will simulate exam conditions, helping students prepare effectively.
- Continuous assessment will track progress, providing regular feedback to guide student improvement.
- Support and feedback will be readily available, with the course instructor offering additional guidance as needed.
- Moodle LMS and interactive boards will ensure easy access to resources and foster an interactive learning environment.
- Video lectures will cover complex topics in detail, allowing students to review content at their own pace.

Textbooks

1. Gonzalez Rafael C. and Woods Richard E., Digital Image Processing, New Delhi: Prentice– Hall of India.
2. Computer Vision: Algorithms and Applications by Richard Szeliski

Online Learning References:

I) **Fast.ai: Practical Deep Learning for Coders**

a. Practical course on deep learning, including applications in image processing and computer vision.

b. Link: Fast.ai - Practical Deep Learning for Coders

II) **Deep Learning for Computer Vision with Python by Adrian Rosebrock**

a. A comprehensive book on deep learning techniques for computer vision applications.

b. Link: PyImageSearch - Deep Learning for Computer Vision with Python

III) **GitHub: OpenCV Projects and Tutorials**

a. Repository of projects and tutorials on OpenCV, a popular library for computer vision.

b. Link: GitHub - OpenCV

IV) **Towards Data Science: Image Processing Tutorials**

a. A collection of tutorials on various image processing techniques and applications.

b. Link: Towards Data Science - Image Processing

V) **IEEE Xplore Digital Library: Image Processing and Computer Vision Papers**

a. Access to research papers and articles on the latest developments in image processing and computer vision.

b. Link: IEEE Xplore Digital Library

IMAGE PROCESSING & COMPUTER VISION LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Image Processing & Computer Vision Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP354	0-0-2	1
Type of Course:	Minor (Department Elective I)		
Pre-requisite(s), if any: (1) Linear Algebra and (2) programming in python			

Defined Course Outcomes

COs	Statements
CO 1	**Applying** image processing techniques using Python libraries.
CO 2	**Analyzing** and evaluate the effectiveness of different image enhancement algorithms.
CO 3	**Implementing** image restoration algorithms and evaluate their performance in the presence of noise.
CO 4	**Developing** image compression algorithms and analyze their impact on image quality.
CO 5	**Formulating** computer vision techniques such as object detection and tracking, gesture recognition, and facial expression recognition using Python.

Lab Experiments

S.N	Experiment	CO
1	Implement a program to acquire and display an image. Demonstrate the process of image sensing and acquisition, and explain the components involved in an image processing system	**CO1**
2	Develop a program to perform sampling and quantization on a given image. Visualize the effects of different sampling rates and quantization levels on image quality.	**CO1**
3	Implement image enhancement techniques in the spatial domain, including contrast stretching and histogram equalization. Compare the results before and after enhancement.	**CO1**
4	Apply frequency domain filtering to an image. Implement both low pass and high pass filters, and demonstrate their effects on the image in terms of noise reduction and edge enhancement.	**CO1**
5	Implement a noise model to simulate different types of noise (Gaussian, salt-and-pepper) in an image. Apply spatial filtering techniques to restore the image from the noisy version.	**CO2**
6	Develop a program to perform periodic noise reduction using frequency domain filtering. Implement and compare inverse filtering and Wiener filtering for image restoration	**CO2**
7	Implement geometric transformations on an image, such as rotation, scaling, and translation. Demonstrate how these transformations affect the image quality and geometry.	**CO2**
8	Perform color image processing by converting an image from RGB to another color model (e.g., HSV, YCbCr). Implement and visualize color-based image segmentation techniques.	**CO2**
9	Implement Huffman Coding and Run Length Coding for image compression. Compare the compression ratios and efficiency of these techniques on different types of images.	**CO3**
10	Develop a program to perform image segmentation using edge	**CO3**

	detection techniques. Implement edge linking and boundary detection algorithms to segment objects within an image.	
11	Apply morphological operations, such as dilation and erosion, on a binary image. Implement basic morphological algorithms to enhance the structure and features of the objects in the image.	**CO3**
12	Implement threshold-based and region-based segmentation techniques. Compare their effectiveness in segmenting different types of images and objects.	**CO3**
13	Implement boundary and regional descriptors to represent the shape and features of objects in an image. Use chain codes and structural methods to describe object	**CO4**
14	Develop a basic convolutional neural network (CNN) for image classification. Train the CNN on a dataset and evaluate its performance in recognizing different classes of images.	CO4
15	Implement a motion estimation algorithm to track moving objects in a video sequence. Demonstrate object tracking by visualizing the trajectories of the tracked objects.	CO4
16	Create a system for face and facial expression recognition using machine learning algorithms. Implement feature extraction techniques and train a classifier to recognize different expressions and identities.	CO4

INTRODUCTION TO GENERATIVE AI

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Introduction to Generative AI	**Course Code**	**L-T-P**	**Credits**
	ENSP306	4-0-0	4
Type of Course:	Minor (Department Elective I)		
Pre-requisite(s), if any:			

Course Perspective. This course provides an in-depth introduction to the principles, techniques, and applications of Generative Artificial Intelligence (AI). Generative AI is a rapidly evolving field that has the potential to transform numerous industries by enabling machines to create content, predict outcomes, and enhance decision-making processes. This course will cover foundational concepts, the latest advancements in generative models, and practical applications, ensuring students gain a comprehensive understanding of the subject.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** generative AI principles, recent advancements, and applications.
CO 2	**Applying** probability theory and statistics in generative AI tasks.
CO 3	**Designing** deep learning models for generative tasks..
CO 4	**Evaluating** generative models for specific applications.

Course Outline:

Unit Number: 1	**Title: Foundations of Generative AI**	**No. of hours: 10**
Content: ☐ **Introduction to Generative AI:** Definition, working principles, and recent advancements. ☐ **Generative Modeling:** • Overview of Generative vs. Discriminative Models. • Introduction to Probabilistic Generative Models. • Naive Bayes as a basic generative model. • Challenges in Generative Modeling. • Introduction to Representation Learning.		
Unit Number: 2	**Title: Deep Learning**	**No. of hours: 10**
Content: ☐ Overview of Structured and Unstructured Data. ☐ Introduction to Deep Neural Networks using Keras and TensorFlow. ☐ Building and Training Deep Neural Networks: • Loading, building, compiling, training, and evaluating models. • Techniques to improve models: Convolutional layers, Batch normalization, and Dropout layers. ☐ Introduction to Autoencoders: • Building and understanding Variational Autoencoders (VAEs). • Using VAEs for tasks like face generation.		

Unit Number: 3	Generative Adversarial Networks	No. of hours: 10
Content: ☐ **Introduction to GANs:** • Roles of the Discriminator and Generator. • Training processes and challenges in GANs. ☐ **Advanced GAN Techniques:** • Wasserstein GAN (WGAN) and WGAN-GP. ☐ **Evaluation of GANs:** • Qualitative and Quantitative methods. ☐ **GAN Architectures and Applications.**.		
Unit Number: 4	**Applications and Future Directions**	**No. of hours: 10**
Content: ☐ **Real-World Applications of Generative AI:** • Image synthesis, data augmentation, healthcare, gaming, and art. ☐ **Ethical Considerations:** • Addressing bias, fairness, deepfakes, and responsible AI practices. ☐ **Emerging Trends:** • Overview of reinforcement learning, meta-learning, and tools like OpenAI's DALL-E.		

Learning Experiences:

- Lecture PPTs will provide clear explanations and visual aids to reinforce key concepts in Generative AI.
- Problem-based theory assignments will encourage students to apply theoretical knowledge to real-world challenges in generative modeling.
- Project-based lab assignments will offer hands-on experience in building and evaluating AI models, making learning practical and interactive.
- Question banks will help students thoroughly review the entire syllabus, ensuring a solid understanding of all topics covered.

- Model question papers will prepare students for exams by familiarizing them with the format and types of questions they may encounter.
- Continuous assessment will track student progress and provide ongoing feedback, helping to identify areas for improvement.
- Support & Feedback will be readily available, with opportunities for one-on-one guidance and peer collaboration to enhance learning outcomes.

Textbooks

1. Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play by David Foster
2. Hands-On Generative Adversarial Networks with Keras: Create Powerful Neural Networks for Real-World Applications by Rafael Valle

References

1. Generative Deep Learning, by David Foster, 2nd Edition, O'Reilly Media, Inc.
2. Deep Learning by Ian Goodfellow, Yoshua Bengio and Aaron Courville, The MIT Press
3. Pattern recognition and machine learning by Christopher M. Bishop

INTRODUCTION TO GENERATIVE AI LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Introduction to Generative AI Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP356	0-0-2	1
Type of Course:	Minor (Department Elective I)		
Pre-requisite(s), if any: NA			

Defined Course Outcomes

COs	
CO 1	**Designing,** develop, and evaluate deep neural networks for complex image classification tasks using advanced model improvement techniques.
CO 2	**Implementing** and analyze variational autoencoders (VAEs) for anomaly detection in network traffic data, focusing on security threat identification.
CO 3	**Developing** and enhance generative adversarial networks (GANs) for realistic image synthesis, emphasizing model architecture and performance improvement.
CO 4	**Utilizing** generative AI techniques for data augmentation in healthcare and other creative fields, assessing the impact on model performance and addressing ethical considerations.

Lab Experiments

S.N	Experiments	COs
1	Implement a basic probabilistic generative model using the Naive Bayes classifier. Train the model on a simple dataset and evaluate its performance. Discuss the challenges faced during the implementation and how they were addressed.	CO1
2	Develop a representation learning model to learn and visualize the latent features of a given dataset. Implement the model using a suitable machine learning framework and analyze the results.	CO1
3	Create a generative modeling framework to generate synthetic data from a given dataset. Compare the performance of generative versus discriminative modeling approaches on the same dataset.	CO1
4	Implement a simple generative model to generate new data points from a probabilistic distribution. Discuss the working principles of generative AI and recent advancements in the field.	CO1
5	Build and train a deep neural network using Keras and TensorFlow to classify images from the MNIST dataset. Evaluate the model's performance and improve it using techniques like convolutional layers, batch normalization, and dropout layers.	CO2
6	Implement an autoencoder and a variational autoencoder (VAE) to compress and reconstruct images from a given dataset. Analyze the differences between the autoencoder and VAE in terms of performance and latent space representation.	CO2
7	Design and implement a convolutional neural network (CNN) to classify images from the CIFAR-10 dataset. Train the model,	CO3

	evaluate its performance, and use techniques like data augmentation to improve its accuracy.	
8	Implement a basic GAN to generate synthetic images. Train the GAN on a simple image dataset, and evaluate the quality of the generated images using both qualitative and quantitative methods.	CO3
9	Develop a Wasserstein GAN (WGAN) and train it on a dataset of images. Compare the performance of the WGAN with a standard GAN in terms of stability and quality of generated images.	CO3
10	Implement and compare different GAN architectures and loss functions. Train each GAN on the same dataset and evaluate their performance using qualitative and quantitative methods.	CO3
11	Implement a generative AI model for image synthesis. Train the model on a dataset of images and evaluate the quality of the synthesized images. Discuss the potential applications of image synthesis in various fields.	CO4
12	Develop a generative AI model for data augmentation and data generation. Train the model on a dataset with limited data and analyze how data augmentation improves the performance of a classifier trained on the augmented dataset.	CO4
13	Create a generative AI application for healthcare, such as generating synthetic medical images for training purposes. Discuss the ethical considerations and challenges associated with using generative AI in healthcare.	CO4

TRANSFER LEARNING

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Transfer Learning	**Course Code**	**L-T-P**	**Credits**
	ENSP308	4-0-0	4
Type of Course:	Minor (Department Elective I)		
Pre-requisite(s), if any:			

Course Perspective. This course introduces students to the fundamental concepts and advanced techniques of transfer learning, an essential area in machine learning and deep learning. Transfer learning focuses on leveraging knowledge gained from one domain to improve learning in another domain. This course covers theoretical foundations, practical implementations, and applications across various domains, providing students with the skills necessary to apply transfer learning to real-world problems.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the theoretical foundations, motivations, and applications of transfer learning.
CO 2	**Implementing** transfer learning algorithms proficiently using Python and deep learning libraries.
CO 3	**Applying** fine-tuning, feature extraction, and model adaptation techniques effectively across different domains and tasks.
CO 4	**Evaluating** transfer learning models using standard metrics and methodologies.

CO 5	**Analyze** real-world applications and ethical implications of transfer learning for inclusive and meaningful solutions.

Course Outline:

Unit Number: 1	**Title: Foundations of Transfer Learning**	**No. of hours: 10**

Content Summary:

Introduction to Transfer Learning:

- Overview of AI, Machine Learning, and Transfer Learning
- Relationship to Existing Machine Learning Paradigms
- Fundamental Research Issues and Applications

Instance-Based Transfer Learning:

- Instance-Based Noninductive Transfer Learning
- Instance-Based Inductive Transfer Learning

Feature-Based Transfer Learning:

- Minimizing Domain Discrepancy
- Learning Universal Features
- Feature Augmentation

Model-Based Transfer Learning:

- Transfer through Shared Model Components
- Transfer through Regularization

Unit Number: 2	**Title: Advanced Transfer Learning Techniques**	**No. of hours: 10**

Content Summary:

Relation-Based Transfer Learning:

- Markov Logic Networks (MLNs)
- Relation-Based Transfer Learning with MLNs

Heterogeneous Transfer Learning:

- Problem Definition and Methodologies

- Applications of Heterogeneous Transfer Learning

Adversarial Transfer Learning:

- Generative Adversarial Networks (GANs)
- Transfer Learning with Adversarial Models

Transfer Learning in Reinforcement Learning:

- Inter-task and Inter-domain Transfer Learning

Unit Number: 3	Title: Multi-task and Theoretical Aspects of Transfer Learning	No. of hours: 12

Content Summary:

Multi-task Learning:

- Supervised, Unsupervised, and Semi-supervised Learning
- Active, Reinforcement, and Online Learning
- Multi-view and Distributed Multi-task Learning

Transfer Learning Theory:

- Generalization Bounds for Multi-task Learning
- Generalization Bounds for Supervised and Unsupervised Transfer Learning

Transitive Transfer Learning (TTL):

- TTL over Mixed Graphs
- TTL with Hidden Feature Representations and Deep Neural Networks

AutoTL: Learning to Transfer Automatically:

- The L2T Framework
- Parameterizing and Inferring What to Transfer
- Connections to Other Learning Paradigms

Unit Number: 4	Title: Applications of Transfer Learning	No. of hours: 8

Content Summary:

Specialized Learning Techniques:

- Few-Shot, Zero-Shot, and One-Shot Learning
- Bayesian Program Learning and Poor Resource Learning

Transfer Learning Applications:

- Computer Vision and Medical Image Analysis
- Natural Language Processing and Sentiment Analysis
- Dialogue Systems and Spoken Language Understanding
- Natural Language Generation

Learning Experiences:

- Lecture PPTs will deliver foundational knowledge on transfer learning, providing clear explanations of key concepts and advanced techniques.
- Problem-based theory assignments will engage students in applying transfer learning concepts to solve real-world problems, enhancing critical thinking.
- Project-based lab assignments will offer hands-on experience with implementing transfer learning models, making learning practical and relevant.
- Question banks will help students review the entire syllabus, reinforcing their understanding of both foundational and advanced transfer learning topics.
- Model question papers will prepare students for exams by familiarizing them with typical question formats and challenges.
- Continuous assessment will provide regular feedback on student progress, helping them identify strengths and areas for improvement.
- Support & Feedback will be available through the course instructor, encouraging collaboration and peer reviews to deepen understanding and improve performance.

Textbook

1. "Transfer Learning" by Qiang Yang, Yu Zhang, Wenyuan Dai, Sinno Jialin Pan

Reference Books:

1. "Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville
2. "Transfer Learning in Action" by Yuxi (Hayden) Liu
3. "Transfer Learning for Natural Language Processing" by Paul Azunre
4. "Deep Learning for Computer Vision" by Rajalingappaa Shanmugamani
5. "Hands-On Transfer Learning with Python" by Dipanjan Sarkar and Raghav Bali

Additional Readings:

I) **Open Source Projects:**

a. TensorFlow Hub: GitHub Repository

b. PyTorch Hub: GitHub Repository

II) **Community Forums and Discussions:**

a. Reddit: r/MachineLearning

b. Stack Overflow: Transfer Learning Tag

III) **OSSU Link:**

Open Source Society University - Data Science Curriculum

TRANSFER LEARNING LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Transfer Learning Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP358	0-0-2	1
Type of Course:	Minor (Department Elective I)		
Pre-requisite(s), if any: NA			

Defined Course Outcomes

COs	Statement
CO 1	**Apply** and evaluate basic transfer learning models and techniques to solve image and text classification tasks using pre-trained models.
CO 2	**Implement** and analyze advanced transfer learning techniques, including relation-based, adversarial, and heterogeneous transfer learning, to improve model performance across different domains.
CO 3	**Develop** and assess multi-task learning models and theoretical aspects of transfer learning, focusing on improving model generalization and performance in multi-domain tasks.
CO 4	**Utilize** transfer learning for practical applications, including few-shot learning, zero-shot learning, sentiment analysis, and privacy-preserving techniques, to address real-world problems effectively.

Lab Experiments

S.N	Experiment	CO
1	Implement an instance-based transfer learning model using a simple dataset to demonstrate the concepts of noninductive and inductive transfer learning.	CO1
2	Develop a feature-based transfer learning model to minimize domain discrepancy and learn universal features. Use a dataset with different domains to showcase the effectiveness of feature augmentation.	CO1
3	Create a model-based transfer learning application by sharing model components and applying regularization techniques. Evaluate the performance of the model on a target task with limited data.	CO1
4	Compare the performance of a naive machine learning model with a transfer learning model on the same dataset. Discuss the advantages and challenges of transfer learning in this context.	CO1
5	Implement a relation-based transfer learning model using Markov Logic Networks (MLNs). Apply the model to a dataset with relational data and evaluate its performance.	CO2
6	Develop a heterogeneous transfer learning application that addresses the challenges of transferring knowledge between different feature spaces or distributions. Demonstrate the application on a real-world dataset.	CO2
7	Create an adversarial transfer learning model using GANs. Train the model on a source domain and evaluate its ability to generate or classify data in a target domain.	CO2
8	Implement transfer learning in a reinforcement learning context by transferring knowledge from one task to another. Compare the performance of the transfer learning model with a model trained from scratch.	CO2
9	Develop a multi-task learning model that simultaneously trains on	CO3

	multiple tasks. Evaluate the performance improvements achieved through shared learning compared to individual task training.	
10	Create a transitive transfer learning model using mixed graphs and hidden feature representations. Apply the model to a dataset with multiple related tasks and analyze the results.	CO3
11	Implement transfer learning models for various applications, such as computer vision, medical image analysis, and natural language processing. Compare the effectiveness of transfer learning across these different domains	CO4

(DEPARTMENT ELECTIVE-II)

COMPUTATIONAL SERVICES IN THE CLOUD

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Computational Services in The Cloud	**Course Code**	**L-T-P**	**Credits**
	ENSP401	4-0-0	4
Type of Course:	Minor (Department Elective II)		
Pre-requisite(s), if any:			

Course Perspective. This course introduces students to the fundamental concepts and applications of cloud computing, exploring the paradigm shift towards cloud-based IT resources and services. It covers various cloud service models (IaaS, PaaS, SaaS), deployment models (public, private, hybrid, community), and key characteristics and challenges of cloud computing. Students will learn about virtualization, microservices, cloud storage, serverless computing, and resource management fundamentals. Additionally, the course includes case studies on cloud market analysis, security, compliance, big data handling, and a comparative study of public clouds. By the end of the course, students will be equipped to understand, implement, and analyze cloud computing technologies and solutions.

Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Explaining** the core concepts of the cloud computing paradigm: how and why this paradigm shift came about, the characteristics, advantages and challenges brought about by the various models and services in cloud computing.

CO 2	**Applying** the fundamental concepts in datacenters to understand the tradeoffs in power, efficiency and cost.
CO 3	**Identifying** resource management fundamentals, i.e. resource abstraction, sharing and sandboxing and outline their role in managing infrastructure in cloud computing.
CO 4	**Analyzing** various cloud programming models and apply them to solve problems on the cloud.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Foundations of Cloud Computing**	**No. of hours: 10**
Content: **Introduction to Cloud Computing:** • Definitions and basic concepts • Cloud delivery models (IaaS, PaaS, SaaS) • Cloud deployment models (Public, Private, Hybrid) • Benefits and challenges of cloud computing **Cloud Infrastructure and Architecture:** • Cloud computing services and inter-cloud interoperability • Virtualization and its importance **Security and Ethical Issues:** • Security and privacy concerns • Ethical issues in cloud computing		
Unit Number: 2	**Title: Advanced Cloud Computing and Virtualization**	**No. of hours: 12**

Content:

Virtualization Technologies:

- Virtual machine monitors
- Full virtualization and paravirtualization
- Virtualization technology (hardware-based and OS-based)

Resource Management and Scheduling:

- Cloud resource management
- Scheduling algorithms and dynamic application scaling
- Optimization of network virtualization

Virtualization Security:

- Virtualization security risks

Unit Number: 3	Title: Cloud Security, Privacy, and Compliance	No. of hours: 10

Content:

Cloud Security Basics:

- Cloud security risks and challenges
- Security mechanisms (encryption, hashing, digital signatures)
- Identity and access management

Advanced Security Measures:

- Trusted virtual machine monitors
- Cloud security policies and controls
- Cloud security threats (traffic eavesdropping, denial of service)

Compliance and Legal Issues:

- Multi-regional compliance
- Privacy impact assessment
- Case studies on cloud security

Unit Number: 4	Title: Applications of Cloud Computing and Future Trends	No. of hours: 8

Content:

Cloud Applications:

- Scientific research and high-performance computing
- Social computing and digital content
- Big data and cloud-based AI/ML applications

Emerging Trends:

- Edge computing and fog computing
- Future challenges and opportunities
- Energy use and ecological impact of data centers

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures and Video Sessions**: Explore cloud computing concepts through engaging presentations.
2. **Problem-Based Theory Assignments**: Analyze real-world scenarios to enhance problem-solving skills.
3. **Project-Based Lab Assignments**: Gain practical experience by implementing cloud solutions in labs.
4. **Collaborative Group Work**: Collaborate on case studies related to cloud services.
5. **Continuous Assessment and Feedback**: Receive ongoing assessments and instructor feedback.

Outside Classroom Learning Experience

1. **Use of ICT Tools and Moodle LMS**: Access course materials anytime via Moodle.
2. **Engagement with a Question Bank and Model Papers**: Prepare for exams with a question bank and model papers.
3. **Application of Theoretical Knowledge to Practical Scenarios**: Apply cloud concepts to real-world situations.

Textbooks:

1. Cloud Computing: Concepts, Technology & Architecture by Thomas
2. "Cloud Computing: Theory and Practice" by Dan C. Marinescu
3. "Cloud Computing: Concepts, Technology & Architecture" by Thomas Erl, Zaigham Mahmood, and Ricardo Puttini

References

1. Lizhe Wang, Rajiv Ranjan, Jinjun Chen and Boualem Benatallah, Cloud Computing (1 ed.), CRC Press, 2017. ISBN 978-1351833097.

2. Judith S. Hurwitz and Daniel Kirsch, Cloud Computing For Dummies (2 ed.), Hoboken: John Wiley & Sons, 2020. ISBN 978-1119546658.

COMPUTATIONAL SERVICES IN THE CLOUD LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Computational Services in The Cloud Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP451	0-0-2	1
Type of Course:	Minor (Department Elective II)		
Pre-requisite(s), if any: Nil			

Defined Course Outcomes

COs	**Statement**
CO 1	**Implementing** advanced resource management and scheduling systems in cloud environments to optimize the efficiency and performance of virtualized resources.
CO 2	**Developing** comprehensive security and compliance frameworks for cloud infrastructures, addressing various security threats and ensuring regulatory compliance.
CO 3	**Enhancing** data privacy and compliance strategies for multi-regional cloud deployments, ensuring adherence to global and regional data protection regulations.
CO 4	**Leveraging** cloud computing resources for high-performance scientific research, enabling scalable and efficient data processing, storage, and analysis.

List of Programs

S.N	Project Detail	COs
1	Set up a virtual machine (VM) on a cloud platform (e.g., AWS, Azure, Google Cloud). Explore different VM configurations and understand the basics of IaaS.	CO1
2	Deploy a simple web application using a PaaS provider (e.g., Heroku, Google App Engine). Demonstrate the deployment process and manage application scaling	CO1
3	Implement a cloud storage solution using a SaaS provider (e.g., Dropbox, Google Drive). Upload, share, and manage files to understand cloud storage benefits and challenges.	CO1
4	Investigate the security and privacy settings of a cloud service provider. Configure security groups and access controls to secure your cloud resources.	CO1
5	Install and configure a virtual machine monitor (VMM) like VMware or VirtualBox. Compare full virtualization and paravirtualization techniques.	CO2
6	Install and configure a virtual machine monitor (VMM) like VMware or VirtualBox. Compare full virtualization and paravirtualization techniques.	CO2
7	Optimize network virtualization by setting up and managing virtual networks in a cloud environment. Analyze the performance benefits of network virtualization.	CO2
8	Implement encryption and hashing mechanisms to secure data stored in the cloud. Demonstrate how these mechanisms protect data integrity and confidentiality.	CO3
9	Set up identity and access management (IAM) in a cloud environment. Configure single sign-on (SSO) for multiple cloud services to streamline user access.	CO3
10	Develop a cloud-based AI application using a cloud provider's machine learning services (e.g., AWS SageMaker, Google AI	CO4

	Platform). Train and deploy a machine learning model in the cloud.	
11	Implement a cloud-based big data solution using Hadoop or Spark on a cloud platform. Process and analyze a large dataset to understand the benefits of cloud-based big data processing.	CO4
12	Explore edge computing by deploying a cloud application that interacts with IoT devices. Demonstrate how edge computing can reduce latency and improve performance.	CO4

MICROSOFT AZURE CLOUD FUNDAMENTALS

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Microsoft Azure Cloud Fundamentals	**Course Code**	**L-T-P**	**Credits**
	ENSP403	4-0-0	4
Type of Course:	Minor (Department Elective II)		
Pre-requisite(s), if any:			

Course Perspective. This course introduces students to the fundamental concepts of cloud computing with a focus on Microsoft Azure. It aims to bridge the gap between theoretical cloud principles and practical Azure applications, emphasizing the relevance of cloud services in modern engineering and technology. Students will explore core topics such as cloud computing models, Azure architecture, compute and networking services, storage services, and cost management. The course is divided into four modules:

1. Introduction to Cloud Computing and Azure Fundamentals
2. Introduction to Microsoft Azure
3. Azure Storage Services and Identity Management
4. Azure Cost Management, Governance, and Monitoring

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Identifying** the core concepts of cloud computing and Microsoft Azure, including deployment models and service models.

CO 2	**Understanding** the benefits of cloud services, such as high availability, scalability, and security.
CO 3	**Understanding** Azure architecture components and compute/networking services, analyzing their functionality and use cases.
CO 4	**Determining** the appropriate Azure storage services for different performance requirements and analyze identity management and security features for access control.
CO 5	**Analyzing** cost management strategies in Azure, analyze governance and compliance tools, and determine effective methods for managing and deploying Azure resources.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Introduction to Cloud Computing and Azure Fundamentals**	**No. of hours: 10**
Content Summary: **Cloud Computing Basics:** What is cloud computing, Delivery models, deployment models, defining attributes, resources, and organization of the infrastructure. Network-Centric Computing and Network-Centric Content. Cloud computing delivery models and services, Applications of cloud, Ethical Issues in Cloud Computing, Major Challenges Faced by Cloud Computing		
Unit Number: 2	**Title Introduction to Microsoft Azure**	**No. of hours: 12**
Content Summary: **Azure Architecture Components**: Azure regions, availability zones, datacentres,		

Azure resources, resource groups, subscriptions, Management groups hierarchy, **Azure Compute and Networking Services:** Compute types comparison: Container instances, VMs, Functions, Virtual machine options: VMs, VM Scale Sets, availability sets, Azure Virtual Desktop, Application hosting options, Virtual networking: Azure Virtual Networks, subnets, peering, DNS, VPN Gateway, ExpressRoute, Public and private endpoints		
Unit Number: 3	**Title: Azure Storage Services and Identity Management**	**No. of hours: 10**
Content Summary: **Azure Storage Services**: Create and manage virtual machines using Azure. Different VM sizes and types based on performance requirements. VM scaling and load balancing for optimizing application performance. Azure storage services: Blob Storage, Table Storage, File Storage, and Disk Storage.		
Unit Number: 4	**Title: Azure Cost Management, Governance, and Monitoring**	**No. of hours: 8**
Content Summary: **Cost Management in Azure:** Factors affecting costs, Pricing and TCO calculators, Azure Cost Management and Billing tool, Tagging usage. **Governance and Compliance:** Azure Blueprints, Azure Policy, Resource locks, Service Trust Portal **Monitoring Tools in Azure:** Azure Advisor, Azure Service Health, Azure Monitor: Log Analytics, alerts, Application Insights		

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures and Video Sessions**: Learn Microsoft Azure concepts through engaging presentations.
2. **Problem-Based Theory Assignments**: Analyze real-world Azure scenarios to develop practical skills.
3. **Project-Based Lab Assignments**: Implement Azure solutions in hands-on labs for experiential learning.

4. **Collaborative Group Work**: Work together on case studies focused on Azure applications.
5. **Continuous Assessment and Feedback**: Receive regular assessments and instructor feedback to monitor progress.

Outside Classroom Learning Experience

1. **Use of ICT Tools and Moodle LMS**: Access course materials anytime via Moodle for flexible learning.
2. **Engagement with a Question Bank and Model Papers**: Prepare for exams using a question bank and model papers.
3. **Application of Theoretical Knowledge to Practical Scenarios**: Apply Azure concepts to real-world scenarios for better understanding.

Text Book:

a) "Cloud Computing: Theory and Practice" by Dan C. Marinescu

b) "Exam Ref AZ-900 Microsoft Azure Fundamentals" by Jim Cheshire.

References

a) "Exam Ref AZ-900 Microsoft Azure Fundamentals" by Jim Cheshire.

b) "Microsoft Azure Essentials: Fundamentals of Azure" by Michael Collier and Robin Shahan.

c) "Azure for Architects: Implementing cloud design, DevOps, IoT, and serverless solutions on your public cloud" by Ritesh Modi.

d) "Azure Security Center: Protecting your cloud workloads" by Yuri Diogenes, Tom Shinder, and Debra Shinder.

e) "Azure Cost Management and Billing" by Sjoukje Zaal

Additional Readings:

Online Learning Resources:

I) **Microsoft Learn:**

Microsoft's official learning platform offers a wide range of Azure courses, modules, and learning paths for beginners to advanced users. Explore topics such as cloud computing basics, Azure architecture components, compute and networking services, storage services, identity management, cost management, governance, and monitoring. Microsoft Learn

R 1. Coursera:

Coursera provides Azure courses offered by top universities and organizations. Topics include cloud computing basics, Azure architecture, services, security, governance, and more. Coursera Azure Courses

2. **Open-Source Society University (OSSU):**

OSSU offers a structured, open-source curriculum for self-learning various topics, including cloud computing and Azure fundamentals. You can follow their curriculum to gain a comprehensive understanding of Azure concepts and services. OSSU Cloud Computing Curriculum

MICROSOFT AZURE CLOUD FUNDAMENTALS LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Microsoft Azure Cloud Fundamentals Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP453	0-0-2	1
Type of Course:	Minor (Department Elective II)		
Pre-requisite(s), if any: Nil			

Lab Experiments

Defined Course Outcomes

COs	**Statement**
CO 1	**Deploying** and manage scalable web applications using Azure architecture components, ensuring high availability, fault tolerance, and optimal performance.
CO 2	**Developing** and optimize Azure storage solutions for data-intensive applications, focusing on efficient data storage, retrieval, and performance.
CO 3	**Establishing** secure and compliant environments in Azure, ensuring governance, cost management, and continuous monitoring for mission-critical applications.
CO 4	**Migrating** on-premise applications to Azure, ensuring minimal downtime and optimized performance through effective planning, resource management, and monitoring..

S.N	Project Detail	COs
1	Set up a cloud environment using a chosen cloud provider (e.g., AWS, Azure, Google Cloud). Explore and document the different delivery models (IaaS, PaaS, SaaS) and deployment models (Public, Private, Hybrid).	CO1
2	Implement a network-centric application using cloud services. Demonstrate how network-centric content can be delivered and managed in a cloud environment.	CO1
3	Explore a cloud-based application (e.g., Google Docs, Office 365). Analyze its benefits and the ethical issues it presents in terms of data privacy and security.	CO1
	Explore Azure regions, availability zones, and datacenters. Create and manage Azure resources and resource groups, and understand the subscription and management groups hierarchy.	CO2
	Compare different Azure compute types (Container instances, VMs, Functions). Create and manage VMs, VM Scale Sets, and availability sets. Explore Azure Virtual Desktop and application hosting options.	CO2
	Set up a virtual network in Azure. Configure subnets, peering, DNS, VPN Gateway, and ExpressRoute. Explore the use of public and private endpoints in Azure networking.	CO2
	Host a web application on Azure using different hosting options. Compare the performance and cost implications of using VMs, Azure App Service, and Azure Functions.	CO2
	Create and manage virtual machines in Azure. Explore different VM sizes and types based on performance requirements. Implement VM scaling and load balancing to optimize application performance.	CO3
	Set up Azure Blob Storage and upload/download data. Explore Table Storage and File Storage services. Implement Disk Storage and understand its use cases.	CO3
	Use the Azure Pricing Calculator and TCO Calculator to estimate the	CO4

	costs of running a sample application on Azure. Explore the Azure Cost Management and Billing tool to monitor and control costs.	
	Set up monitoring for an Azure application using Azure Monitor. Configure Log Analytics, set up alerts, and use Application Insights to monitor application performance and health	CO4
	Use Azure Advisor and Azure Service Health to optimize and maintain the health of Azure resources. Implement recommendations provided by Azure Advisor and monitor service issues using Azure Service Health.	CO4

STORAGE AND DATABASES ON CLOUD

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Storages and Databases on Cloud**	**Course Code**	**L-T-P**	**Credits**
	ENSP405	4-0-0	4
Type of Course:	Minor (Department Elective II)		
Pre-requisite(s), if any:			

Course Perspective. The course covers the basics of cloud computing and introduces various cloud storage and database types. It discusses migration techniques, security, and performance considerations for cloud databases. The AWS cloud storage unit focuses on Amazon S3, EC2 Instance Storage, and more. It also help student analyzing case studies of companies like Netflix and Spotify using cloud storage and databases.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** cloud storage and database fundamentals, including security best practices.
CO 2	**Applying** indexing, caching, and query optimization for performance in cloud storage and databases.
CO 3	**Analyzing** requirements to select suitable cloud storage and database solutions.
CO 4	**Differentiating** between types of cloud storage and database services.

CO 5	**Articulating** best practices for designing scalable, reliable, and secure cloud storage and databases.

Course Outline:

Unit Number:1	**Title: Introduction to Storage on cloud**	**No. of hours: 8**
Content: Introduction to Cloud Computing, Overview of cloud databases and cloud storages, types of cloud storages (Object, block and file), different types of cloud database management systems, Gartner Magic Quadrant for Cloud Database Management Systems, Advantages of Working with Cloud Databases, Considerations for Cloud Databases, Top Cloud Database, Factors that help in choosing the right cloud database, Challenges involved in using cloud storages and databases.		
Unit Number:2	**Title: Data Integration, Migration, Security and performance on cloud NET FrameworkFundamentals**	**No. of hours: 10**
Content: Techniques, tools, methods, and considerations for migrating from on-premise databases to cloud databases; backup, recovery, and disaster planning, including automated backups, point-in-time recovery, and replication; performance optimization and monitoring, including query optimization, indexing, caching, and monitoring tools; scalability and high availability, including load balancing, replication, sharding, and auto-scaling; cloud data warehousing.		
Unit Number:3	**Title: Cloud-Hosted Data Storage Systems**	**No. of hours: 10**
Content: **Introduction,** Introduction to AWS cloud storage, AWS management console, AWS Storage Services, Uploading files and images, Creating a web server, Overview of Amazon		

S3, Storage Classes, EC2 Instance Storage, network file system Amazon Elastic Block Store, Amazon Elastic file system, Amazon Cloud Front. Brief introduction to Google Cloud Storage, and Azure Blob Storage.		
Unit Number:4	**Title: Case Study**	**No. of hours: 12**
Content: Case Studies and Real-world Examples of Netflix , Airbnb, Pinterest, spotify, coca- cola etc. Analyzing real-world use cases of organizations using cloud storage and databases, discussing architecture decisions, challenges, and lessons learned.		

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Learn cloud storage and database concepts through engaging presentations.
2. **Problem-Based Assignments**: Analyze real-world scenarios to improve understanding of cloud solutions.
3. **Project Labs**: Implement cloud database solutions in hands-on labs.
4. **Collaborative Work**: Team up on case studies related to cloud storage management.
5. **Continuous Feedback**: Receive regular assessments and instructor feedback.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle.
2. **Question Bank**: Use a question bank and model papers for exam preparation.
3. **Real-World Applications**: Apply cloud concepts to practical scenarios.

References

1. Cloud Computing: Concepts, Technology & Architecture by Thomas Erl, Ricardo Puttini, and Zaigham Mahmood
2. Designing Data-Intensive Applications by Martin Kleppmann

3. Cloud Architecture Patterns: Using Microsoft Azure" by Bill Wilder

Additional Readings:

Online Learning Resources for Storage and Databases on Cloud

I) **Microsoft Learn: Introduction to Azure Storage**

a. **Description:** Comprehensive learning path covering Azure Storage services, including Blob, File, and Disk Storage.

b. **Link:** Microsoft Learn - Introduction to Azure Storage

II) **AWS Training and Certification: Storage Learning Path**

a. **Description:** AWS offers a detailed learning path for storage services, including Amazon S3, EBS, and more.

b. **Link:** AWS Training - Storage Learning Path

III) **Google Cloud Training: Storage and Databases**

a. **Description:** Google Cloud offers courses on Cloud Storage, SQL, and NoSQL database services.

b. **Link:** Google Cloud Training - Storage and Databases

STORAGE AND DATABASES ON CLOUD LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Storages and Databases on Cloud Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP455	0-0-2	1
Type of Course:	Minor (Department Elective II)		
Pre-requisite(s), if any: Nil			

Defined Course Outcomes

COs	Statements
CO 1	**Implementing** database migration, backup, recovery, and performance optimization strategies for transitioning on-premise databases to AWS cloud.
CO 2	**Developing** cloud storage solutions for large-scale file management and optimize performance using AWS storage services and content delivery networks.
CO 3	**Designing** and manage cloud data warehousing solutions, including ETL processes, performance monitoring, and scalability configurations.
CO 4	**Analyzing** and apply best practices from real-world cloud storage use cases to enhance the scalability, reliability, and performance of cloud-based applications.

Lab Experiments

Project No.	Project Detail	Mapped CO/COs
1	Explore different types of cloud storages (Object, Block, File). Set up and compare examples of each type using a cloud provider (e.g., AWS S3 for object storage, EBS for block storage, EFS for file storage).	CO1
2	Research and analyze the Gartner Magic Quadrant for Cloud Database Management Systems. Create a report summarizing the top cloud database providers and their key features.	CO1
3	Implement a migration process from an on-premise database to a cloud database using a migration tool (e.g., AWS Database Migration Service, Google Cloud Database Migration Service). Document the steps and considerations involved.	CO2
4	Develop a cloud storage solution for a media sharing platform using AWS storage services to handle large-scale file uploads and downloads.	CO2
5	Configure Amazon CloudFront for content delivery. Upload and distribute content using CloudFront and analyze the performance benefits. Briefly explore and set up storage using Google Cloud Storage and Azure Blob Storage.	CO3
6	Create a cloud data warehouse for an e-commerce company to store and analyze sales data using AWS Redshift.	CO3
7	Develop a cloud storage and content delivery network (CDN) solution for a video streaming service to ensure	CO4
8	Conduct a comprehensive analysis of how major companies like Netflix, Airbnb, and Spotify use cloud storage and databases to enhance their operations.	CO4

APPLICATION DEVELOPMENT AND DEVOPS ON CLOUD

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: APPLICATION DEVELOPMENT AND DEVOPS ON CLOUD	**Course Code**	**L-T-P**	**Credits**
	ENSP407	4-0-0	4
Type of Course:	Minor (Department Elective II)		
Pre-requisite(s), if any: Nil			

Course Perspective. The syllabus aims to equip students with practical skills and theoretical knowledge to design, develop, and deploy applications in cloud environments while implementing DevOps practices to enhance software development, delivery, and operations on the cloud. It prepares them for a career in the dynamic and rapidly growing field of cloud computing and DevOps, where demand for skilled professionals is high due to the increasing adoption of cloud technologies in various industries.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the fundamental concepts of cloud computing and the various service and deployment models.
CO 2	**Developing** cloud-native applications using containerization and microservices architecture.
CO 3	**Implementing** DevOps practices in cloud environments, including CI/CD pipelines and Infrastructure as Code.

CO 4	**Utilizing** cloud-based DevOps tools for version control, collaboration, testing, and performance optimization.
CO 5	**Analyzing** best practices for application security, cost management, and high availability in the cloud.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: The problem of Delivering Software**	**No. of hours: 10**
Content Summary: Introduction to DevOps: Principles, Practices, Common Release antipatterns, benefits. Configuration Management: using version control, managing dependencies, managing software configuration, managing tools		
Unit Number: 2	**Title: Continuous Integration and Testing Strategy**	**No. of hours: 10**
Content Summary: Introduction to contigous integration. Implementing contiguous integration, Essential practices, distributed version control system. Testing Strategy: Introduction of testing, Types of tests, real-life situation and strategies, and managing defect backlogs		
Unit Number: 3	**Title: The deployment Pipeline**	**No. of hours: 10**
Content Summary: Anatomy of the Deployment Pipeline: Introduction of deployment pipeline, deployment pipeline practices, the automated acceptance test gate, test strategy, prepare to release, implement a deployment pipeline. Build and deployment scripting, the commit stage: principles and practices, Automated Acceptance		

testing, Testing Non functional Requirements, deploying and releasing application.		
Unit Number: 4	**Title: The delivering Ecosystem**	**No. of hours: 10**
Content Summary: Managing infrastructure and Environments: understanding the needs of the operation team . Managing server provisioning and configuration, managing the configuration of middleware, managing infrastructure services, virtualization, cloud architecture, monitoring infrastructure and application, managing data: Database scripting, data management and deployment pipeline. Managing components and dependencies: Introduction, keeping your application releasable, dependencies, components, managing dependency graph. Managing Continuous delivery: introduction, maturity model, project lifecycle, risk management process.		

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Learn application development and DevOps concepts through engaging presentations.
2. **Problem-Based Assignments**: Analyze real-world scenarios to enhance development and deployment skills.
3. **Project Labs**: Implement cloud-based applications and DevOps practices in hands-on labs.
4. **Collaborative Work**: Work in teams on case studies related to application deployment.
5. **Continuous Feedback**: Receive regular assessments and instructor feedback to monitor progress.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle for flexible learning.
2. **Question Bank**: Utilize a question bank and model papers for effective exam preparation.

3. **Real-World Applications**: Apply development and DevOps concepts to practical scenarios.

Textbooks:

- Jez Humble and David Farley, Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, Pearson Education, Inc., 2011.

References

- Jez Humble and David Farley, Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, Pearson Education, Inc., 2011.
- Thomas Erl, Ricardo Puttini, and Zaigham Mahmood, Cloud Computing: Concepts, Technology & Architecture, Prentice Hall, 2013.
- Arun Eapen, Docker on Amazon Web Services: Build, deploy, and manage your container applications at scale on AWS, Packt Publishing, 2017.
- Sam Newman, Building Microservices: Designing Fine-Grained Systems, O'Reilly Media, Inc., 2015.
- Mark Richards and Neal Ford, Fundamentals of Software Architecture: An Engineering Approach, O'Reilly Media, Inc., 2020.

Additional Readings:

Online Learning Resources for Application Development and DevOps on Cloud

I) **Microsoft Learn: Azure DevOps and Development**

a. **Description:** Comprehensive learning paths and modules on Azure DevOps, including CI/CD, IaC, and cloud-based application development.

b. **Link:** Microsoft Learn - Azure DevOps and Development

II) **AWS Training and Certification: DevOps on AWS**

a. **Description:** Detailed courses and certifications for learning DevOps practices and application development on AWS, covering tools like AWS CodePipeline, CodeBuild, and more.

b. **Link:** AWS Training - DevOps on AWS

III) **Google Cloud Training: Application Development**

a. **Description:** Google Cloud provides courses on developing applications using Google Cloud services, including Kubernetes, App Engine, and Cloud Functions.

b. **Link:** Google Cloud Training - Application Development

APPLICATION DEVELOPMENT AND DEVOPS ON CLOUD LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: APPLICATION DEVELOPMENT AND DEVOPS ON CLOUD LAB	**Course Code**	**L-T-P**	**Credits**
	ENSP457	0-0-2	1
Type of Course:	Minor (Department Elective II)		
Pre-requisite(s), if any: Nil			

Lab Experiments

Defined Course Outcomes

COs	Course Outcomes (COs)
CO 1	**Implementing** continuous integration (CI) pipelines to automate the build, test, and integration processes, ensuring smooth and efficient integration of new code changes.
CO 2	**Developing** and implement automated deployment pipelines for microservices and mobile applications, ensuring reliable and efficient deployment processes.
CO 3	**Integrating** comprehensive testing strategies, including acceptance and non-functional requirements testing, into CI/CD pipelines to ensure high code quality and performance standards.
CO 4	**Managing** and monitor cloud-based application infrastructure using automation tools, ensuring efficient provisioning, configuration, and continuous monitoring.

S.N	Experiment	COs
1	Set up a version control system (e.g., Git) for a sample software project. Demonstrate how to manage code versions, branches, and merges.	CO1
2	Implement configuration management using a tool such as Ansible or Chef. Create scripts to manage software configurations and dependencies for a sample application.	CO1
3	Explore common release antipatterns in software delivery. Analyze a real-world case study and propose solutions to mitigate these antipatterns using DevOps principles.	CO1
4	Implement continuous integration for a sample project using a CI tool (e.g., Jenkins, Travis CI). Configure the tool to automatically build and test the project whenever code changes are committed.	CO2
5	Set up a distributed version control system (e.g., Git) for a collaborative project. Demonstrate branching, merging, and managing code changes in a distributed environment.	CO2
6	Implement a deployment pipeline for a sample application. Automate the build, test, and deployment stages using a CI/CD tool like Jenkins or GitLab CI.	CO3
7	Write and execute build and deployment scripts for a sample project. Use scripting languages like Bash or PowerShell to automate the process.	CO3
8	Set up and configure infrastructure for a sample application using Infrastructure as Code (IaC) tools like Terraform or CloudFormation. Demonstrate server provisioning, middleware configuration, and monitoring.	CO4
9	Implement continuous delivery for a sample project. Develop a maturity model, define the project lifecycle, and establish a risk management process to ensure smooth delivery and deployment.	CO4

(DEPARTMENT ELECTIVE-III)

MOBILE APPLICATION DEVELOPMENT USING IOS

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Mobile Application Development using iOS	**Course Code**	**L-T-P**	**Credits**
	ENSP409	4-0-0	4
Type of Course:	Minor (Department Elective III)		
Pre-requisite(s), if any: Basics of Android			

Course Perspective. The objective of the course is to provide skills to develop applications for OS X and iOS. It includes an introduction to the development framework Xcode. Objective-C is used as a programming language to develop applications. Objective-C is the superset of the C programming language and provides object-oriented capabilities and a dynamic runtime. Objective-C inherits the syntax, primitive types, and flow control statements of C and adds syntax for defining classes and methods. The course is divided into 4 modules:

1. Introduction to IDE and SDK of iOS App Development
2. Swift Programming
3. Encapsulating Data
4. Developing iOS Applications

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the fundamental concepts of variables, constants, and basic data types in SWIFT.

CO 2	**Analyzing** the use of control flow statements such as for, if, and switch in various programming scenarios.
CO 3	**Applying** object-oriented concepts in SWIFT, including the use of classes, structures, and protocols.
CO 4	**Creating** functions, closures, and extensions to enhance code modularity and reuse.
CO5	**Evaluating** error handling techniques and type checking mechanisms to develop robust SWIFT applications

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Introduction to SWIFT Language**	**No. of hours: 10**
Content Summary: Variables & Constants, Introduction to functions (methods), Arrays, Dictionaries, Data, Date and other basic data types, Enums, structures, closuresFor, If, switch statement, Object oriented concepts with SWIFT, Type check, AnyObject, Any Protocols, Extensions, Error handling, Working with classes		
Unit Number: 2	**Title: Working with Xcode**	**No. of hours: 8**
Content Summary: Introduction to XCODE, COCOA touch framework, iOS application architecture, Application lifecycle		
Unit Number: 3	**Title: Introduction to view controllers and Views**	**No. of hours: 12**

Content Summary: View Controllers, view, view lifecycle, Basic Controls – Label, Buttons, Text field, image View, Table view with default cells and customized cells, Collection view with default cells and customized cells, Picker view, Date picker, scroll view, navigation and Tab bar controller, Understanding Interface builder, XIB files, Creating outlets and Actions, Handling touch and gesture events, Segment and Page control, switch view, UIAlertView		
Unit Number: 4	**Title: Integrating with Database**	**No. of hours: 10**
Content Summary: Introduction to data storage methods in iOS, Using Core Data, SQLite database, User Defaults, Property List		

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Explore iOS app development concepts through engaging presentations.
2. **Problem-Based Assignments**: Analyze real-world scenarios to improve mobile development skills.
3. **Project Labs**: Build and test iOS applications in hands-on lab sessions.
4. **Collaborative Work**: Work in teams on case studies related to mobile app development.
5. **Continuous Feedback**: Receive regular assessments and instructor feedback to track progress.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle for convenient learning.
2. **Question Bank**: Use a question bank and model papers for exam preparation.
3. **Real-World Applications**: Apply iOS development concepts to practical mobile app scenarios.

References

1. iOS 14 Programming for Beginners: Kickstart your iOS app development journey with the Swift programming language and Xcode 12, 6th Edition, Ahmad Sahar and Craig Clayton.
2. Mastering iOS 14 Programming: Build professional-grade iOS applications with Swift 5 and Xcode 12.

Additional Readings:

Online Learning Resources for Mobile Application Development Using iOS

- **Apple Developer Documentation**
  - **Description:** Comprehensive documentation and tutorials for iOS app development using Swift and Xcode.
  - **Link:** Apple Developer Documentation
- **Ray Wenderlich: iOS and Swift Tutorials**
  - **Description:** A collection of high-quality tutorials and courses on iOS app development, covering Swift, Xcode, and various iOS frameworks.
  - **Link:** Ray Wenderlich iOS Tutorials
- **GitHub: iOS Development Resources**
  - **Description:** A curated list of open-source projects, libraries, and resources for learning and improving iOS development skills.
  - **Link:** GitHub - iOS Development Resources

MOBILE APPLICATION DEVELOPMENT USING IOS LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Mobile Application Development using iOS Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP459	0-0-2	1
Type of Course:	Minor (Department Elective III)		
Pre-requisite(s), if any: Basics of Android			

Lab Experiments

Defined Course Outcomes

COs	
CO 1	**Understanding** and apply fundamental concepts of iOS development using Xcode and the Cocoa Touch framework to build robust and user-friendly applications.
CO 2	**Developing** interactive and dynamic user interfaces in iOS applications using view controllers, views, and gesture recognizers.
CO 3	**Creating** and manage user interfaces and view controllers in iOS applications using Xcode, demonstrating proficiency in Interface Builder and UI Kit components.
CO 4	**Developing** interactive and dynamic user interfaces in iOS applications using view controllers, views, and gesture recognizers.

S.N	Experiment	COs
1	Set up the iOS development environment by installing Xcode. Create a simple "Hello, World!" iOS application to familiarize with the Xcode IDE and Swift programming basics.	CO1
2	Develop a basic iOS application that demonstrates the use of Swift syntax, variables, data types, and control flow. Create a simple calculator app to perform basic arithmetic operations.	CO1
3	Use Xcode and Interface Builder to design a user interface for an iOS app. Create a simple user interface with labels, buttons, and text fields, and handle user interactions.	CO1
4	Implement a simple iOS app to demonstrate the app lifecycle and navigation between view controllers. Create a multi-screen app that navigates between different views using navigation controllers.	CO1
5	Design a responsive user interface using Auto Layout and the constraint system. Create an iOS app with a login screen that adjusts to different screen sizes and orientations.	CO2
6	Implement navigation between different views using storyboards and segues. Create a multi-screen app with a main	CO2

	menu and detailed views for each menu item.	
7	Implement gesture recognition and touch event handling in an iOS app. Create an app that responds to tap, swipe, and pinch gestures to perform different actions.	CO2
8	Implement data persistence using Core Data. Create an iOS app that allows users to add, edit, and delete notes, and save them to a local database.	CO3
9	Use User Defaults and the file system to store and retrieve user preferences and data. Create an app that saves user settings and displays them when the app is reopened.	CO3
10	Implement offline data storage and synchronization. Create an iOS app that allows users to add data while offline and syncs with a remote server when the device is back online.	CO3
11	Implement advanced UI components and animations in an iOS app. Create a visually appealing app with custom views, animations, and transitions between screens.	CO4
12	Access and use iOS sensors and hardware features. Create an app that uses the camera to take photos, and the GPS to display the user's current location on a map.	CO4
13	Debug and test an iOS app using Xcode's debugging tools. Implement unit tests and UI tests to ensure the app functions correctly under different scenarios.	CO4

DEVOPS & AUTOMATION

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **DevOps & Automation**	**Course Code**	**L-T-P**	**Credits**
	ENSP411	4-0-0	4
Type of Course:	Minor (Department Elective III)		
Pre-requisite(s), if any: Nil			

Course Perspective. Throughout the subject, students will engage in hands-on exercises and projects to gain practical experience with various DevOps tools and practices. By the end of the course, students will be well-equipped to embrace the DevOps culture and apply automation techniques to enhance software development, delivery, and operations processes.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the principles and benefits of DevOps, and its role in enhancing collaboration and efficiency between development and operations teams.
CO 2	**Acquiring** hands-on experience with popular DevOps tools such as Git, Jenkins, Docker, Kubernetes, and Ansible for implementing continuous integration, continuous delivery, and automated

	deployment processes.
CO 3	**Demonstrating** proficiency in containerization and orchestration techniques using Docker and Kubernetes for efficient and scalable application deployment and management.
CO 4	**Implementing** configuration management and Infrastructure as Code (IaC) using Ansible and Terraform to automate the provisioning and management of infrastructure resources.
CO 5	**Developing** skills in monitoring, logging, and security practices in the context of DevOps, ensuring application performance, resilience, and adherence to security best practices.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Introduction to DevOps**	**No. of hours: 12**

Content Summary:		
DevOps Principles and Culture: Understand the core principles of DevOps and its cultural impact. Collaboration, automation, continuous integration, continuous delivery, and continuous deployment. **DevOps Toolchain**: Overview of tools and technologies used in DevOps practices. Introduction to popular DevOps tools like Git, Jenkins, Docker, Kubernetes, and Ansible. **Version Control with Git**: Branching, merging, and collaborative development using Git. **Continuous Integration (CI):** Setting up CI pipelines with Jenkins for automated building and testing. **Continuous Delivery and Deployment:** Implementing CD pipelines for deploying.		
Unit Number: 2	**Title: Version Control and CI/CD**	**No. of hours: 8**

Content Summary: **Version Control with Git**: Version control concepts, Git workflows, and collaboration strategies. **Continuous Integration with Jenkins**: Setting up Jenkins pipelines, automated testing, and deployment. **Maven Integration**: Integrate Maven for dependency management and building projects.		
Unit Number: 3	**Title: Containerization and Orchestration**	**No. of hours: 8**
Content Summary: **Introduction to Docker:** Docker concepts, container management, and Docker file creation. **Container Orchestration with Kubernetes:** Kubernetes architecture, deployment, scaling, and networking. **Docker Compose:** Managing multi-container applications with Docker Compose.		
Unit Number: 4	**Title: Configuration Management and Monitoring**	**No. of hours: 12**
Content Summary: **Configuration Management with Ansible**: Ansible playbooks, roles, and infrastructure automation. **Infrastructure as Code (IaC)**: Terraform for provisioning and managing infrastructure. **Monitoring and Logging**: Monitoring tools, log management, and application performance monitoring in DevOps. **Security in DevOps:** Implementing security best practices in CI/CD pipelines and containerized environments.		

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Explore DevOps concepts and automation techniques through engaging presentations.
2. **Problem-Based Assignments**: Analyze real-world scenarios to enhance DevOps and automation skills.
3. **Project Labs**: Implement automation tools and practices in hands-on lab sessions.
4. **Collaborative Work**: Work in teams on case studies related to DevOps implementation.
5. **Continuous Feedback**: Receive regular assessments and instructor feedback to monitor progress.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle for flexible learning.
2. **Question Bank**: Utilize a question bank and model papers for effective exam preparation.
3. **Real-World Applications**: Apply DevOps and automation concepts to practical scenarios.

References

- Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, Authors: Jez Humble and David Farley, Publisher: Pearson Education, Inc., Year: 2011
- The Kubernetes Book, Author: Nigel Poulton, Publisher: Independently published, Year: 2018
- Building Microservices: Designing Fine-Grained Systems, Author: Sam Newman, Publisher: O'Reilly Media, Inc., Year: 2015
- Microservices Patterns: With examples in Java, Author: Eberhard Wolff, Publisher: Manning Publications, Year: 2018
- Terraform: Up & Running: Writing Infrastructure as Code, Author: Yevgeniy Brikman, Publisher: O'Reilly Media, Inc., Year: 2017

Additional Readings:

Online Learning Resources for DevOps & Automation

I) **Kubernetes Academy by VMware**

a. **Description:** Free courses provided by VMware on Kubernetes, covering everything from basic concepts to advanced orchestration techniques.

b. **Link:** Kubernetes Academy by VMware

II) **HashiCorp Learn: Terraform**

a. **Description:** HashiCorp's official resource for learning Terraform, providing tutorials and hands-on labs for infrastructure as code.

b. **Link:** HashiCorp Learn - Terraform

III) **Docker: Docker for Developers**

a. **Description:** Docker's official training resources for developers, covering containerization, Docker Compose, and more.

b. **Link:** Docker - Docker for Developers

DEVOPS & AUTOMATION LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **DevOps & Automation Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP461	0-0-2	1
Type of Course:	Minor (Department Elective III)		
Pre-requisite(s), if any:			

Lab Experiments

Defined Course Outcomes

COs	Course Outcomes
CO 1	**Implementing** collaborative development and continuous integration using Git and Jenkins, demonstrating proficiency in version control, automated testing, and deployment processes.
CO 2	**Developing** and deploy microservices applications using Docker for containerization and Kubernetes for orchestration, managing multi-container applications efficiently.
CO 3	**Managing** automated infrastructure provisioning and configuration using Ansible and Terraform, demonstrating expertise in infrastructure as code and configuration management.
CO 4	**Implementing** continuous monitoring, logging, and security best practices in a DevOps environment, ensuring application performance, system health, and data integrity.

S.N	Experiment	Mapped CO(s)
1	Set up a Git repository and practice branching, merging, and collaborative development. Create a small project and manage code versions using Git.	CO1
2	Install and configure Jenkins for continuous integration. Create a simple CI pipeline that automatically builds and tests a project whenever code changes are committed to the repository.	CO1
3	Implement a continuous delivery pipeline using Jenkins. Deploy a sample application to a staging environment automatically after successful builds and tests.	CO1
4	Implement different Git workflows (e.g., GitFlow, Feature Branch Workflow) for a collaborative project. Manage branches, merges, and resolve conflicts.	CO2
5	Set up a Jenkins pipeline for continuous integration. Configure automated testing and deployment for a sample project. Integrate with a version control system like Git.	CO2
6	nstall Docker and create Dockerfiles for a sample application. Build, run, and manage containers using Docker commands.	CO3
7	Use Docker Compose to manage multi-container applications. Create a Docker Compose file to run a web application with a database and other services.	CO3
8	Use Terraform to provision and manage cloud infrastructure. Create Terraform scripts to deploy a web application on a cloud provider (e.g., AWS, Azure).	CO4
9	Set up monitoring and logging for a sample application. Use tools like Prometheus, Grafana, and ELK Stack (Elasticsearch, Logstash, Kibana) to monitor and analyze application performance and logs.	CO4

.NET FRAMEWORK

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **.NET Framework**	**Course Code**	**L-T-P**	**Credits**
	ENSP413	4-0-0	4
Type of Course:	Minor (Department Elective III)		
Pre-requisite(s), if any:			

Course Perspective. The ".NET Framework" syllabus covers introduction and components of .NET, programming languages, Visual Studio, OOP, exception handling, memory management, Windows Forms/WPF, ASP.NET, web services, .NET Core, Entity Framework, and WCF. Emphasis on practical application and development skills for building robust and secure applications.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** .NET Framework's architecture, CLR, and CTS for cross-language integration and platform independence.
CO 2	**Applying** OOP concepts in .NET for designing robust software solutions.
CO 3	**Utilizing** Visual Studio debugging for diagnosing and fixing errors in .NET applications.
CO 4	**Demonstrating** proficiency in memory management and garbage collection in .NET.
CO 5	**Designing** web applications using ASP.NET, incorporating best practices.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number:1	**Title: .NETFramework**	**No. of hours: 8**
Content Summary: NET Framework - Architecture, Common Language Runtime, Common Type System, Namespaces, Assemblies, Memory Management, Process Management, Class Libraries		
Unit Number:2	**Title:.NET Framework Fundamentals**	**No. of hours: 8**
Content Summary: Object-Oriented Programming (OOP) in .NET, Classes, objects, and inheritance, Exception Handling and Debugging, Debugging techniques and tools in Visual Studio, Logging and error reporting in .NET applications, Memory Management and Garbage Collection, Automatic memory management in .NET, Garbage collection, Finalizers and the Dispose pattern		
Unit Number:3	**Title: Building Applications with .NET Framework**	**No. of hours: 12**
Content Summary: .NET - Declaration, Expression, Control Structures, Function, String, Array, Encapsulation, Class, Property, Indexer, Delegate, Inheritance, Interface, Polymorphism, Exception Handling, Modules, Graphics, File handling and Data Access. .NET – Form- Event–Form Controls – Containers – Menus - Data controls - Printing – Reporting – Dialogs – Components - Single and Multiple Document Interfaces.		
Unit Number:4	**Title: ASP.NETFramework**	**No. of hours: 12**

Content Summary:

ASP.NET – Web Pages, Web Forms, Web Site Design, Data Controls, Validation Controls, HTML, Navigation Controls, Login Controls, Reports - Master Pages – Web Service Architecture - Basic Web Services – Web Reference – Standards

Learning Experiences:

1. **Interactive Lectures**: Explore .NET Framework concepts through engaging presentations.
2. **Problem-Based Assignments**: Analyze real-world scenarios to enhance .NET development skills.
3. **Project Labs**: Build and test applications using the .NET Framework in hands-on labs.
4. **Collaborative Work**: Work in teams on case studies related to .NET application development.
5. **Continuous Feedback**: Receive regular assessments and instructor feedback to track progress.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle for convenient learning.
2. **Question Bank**: Utilize a question bank and model papers for effective exam preparation.
3. **Real-World Applications**: Apply .NET concepts to practical application scenarios.

Textbooks

1. Pro C# 8 with .NET Core: Foundational Principles and Practices in Programming by Andrew Troelsen and Philip Japikse, Apress, 9th Edition, 2020
2. Pro ASP.NET Core 3 by Adam Freeman, Apress
3. ASP.NET Core in Action by Andrew Lock

Additional Readings:

Online Learning Resources:

I) Online Tutorials and Documentation: Direct students to the official Microsoft

documentation for .NET Framework, which provides comprehensive guides and resources. Microsoft .NET Documentation

II) Hands-on Coding Exercises: Assign coding exercises from platforms like LeetCode or HackerRank that focus on implementing concepts of .NET Framework. LeetCode HackerRank

III) Project-Based Learning: Encourage students to work on small projects using different aspects of the .NET Framework. Provide examples of project ideas and resources like GitHub repositories for inspiration. GitHub

.NET FRAMEWORK LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **.NET Framework Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP463	0-0-2	1
Type of Course:	Minor (Department Elective III)		
Pre-requisite(s), if any: Nil			

Defined Course Outcomes

COs	Statements
CO 1	**Understanding** and apply object-oriented design principles, exception handling, memory management, and debugging techniques to develop robust .NET applications.
CO 2	**Developing** graphical user interfaces and handle events in .NET applications to create interactive and user-friendly software solutions.
CO 3	**Implementing** web development techniques in ASP.NET, including web forms, user authentication, master pages, and web services to build secure and dynamic web applications.
CO 4	**Analyzing** and utilize data handling, reporting, and visualization techniques to create comprehensive and functional software systems for various domains.

Lab Experiments

S.N	Experiment	Mapped CO/COs
1	Explore the architecture of the .NET Framework. Create a simple console application to understand the basic structure and components of a .NET project.	CO1
2	Demonstrate the functionality of the Common Language Runtime (CLR). Create a .NET application that uses various data types and namespaces to show how the CLR manages execution.	CO1
3	Implement a .NET application that showcases the Common Type System (CTS). Define and use various data types, and demonstrate type conversion and interoperability.	CO1
4	Create and manage assemblies in a .NET application. Demonstrate how to build, reference, and use assemblies in a multi-project solution.	CO1
5	Implement a simple object-oriented application in .NET. Define classes, create objects, and demonstrate inheritance and polymorphism.	CO2
6	Implement a .NET application that logs errors and handles exceptions gracefully. Use a logging framework (e.g., NLog, log4net) to record application events and errors.	CO2
7	Build a .NET application demonstrating advanced OOP concepts such as encapsulation, properties, indexers, delegates, interfaces, and polymorphism.	CO3
8	Create a .NET application that handles graphics and file I/O. Implement functionality to draw shapes, handle images, and perform file read/write operations.	CO3
9	Implement a .NET application with a rich user interface. Use	CO3

	forms, event handling, form controls, containers, menus, data controls, printing, and reporting functionalities to create a feature-rich application.	
10	Create a basic ASP.NET web application. Design web pages and web forms to understand the structure and components of an ASP.NET project.	CO4
11	Develop an ASP.NET application with user authentication. Use login controls to implement user authentication and authorization, and create a simple reporting feature to display user data.	CO4
12	Create and consume a basic web service in ASP.NET. Implement a web service that provides data to a client application, and demonstrate how to use web references to integrate the web service with an ASP.NET project.	CO4

NEW AGE PROGRAMMING LANGUAGES

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **New-Age programming languages**	**Course Code**	**L-T-P**	**Credits**
	ENSP415	4-0-0	4
Type of Course:	Minor (Department Elective III)		
Pre-requisite(s), if any: Nil			

Course Perspective. New-Age programming languages (GO, F#, Clojure, Kotlin) provides an introduction to the concepts and applications of modern programming languages. It explore the features and benefits of GO, F#, Clojure, and Kotlin, and develop practical skills in programming using these languages. The course will cover language syntax, data types, control structures, functional programming concepts, concurrency, and integration with other technologies.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** principles and paradigms of modern programming languages.
CO 2	**Developing** proficiency in syntax, data structures, and control flow of each language.
CO 3	**Exploring** unique features and strengths of each language.
CO 4	**Applying** development tools to improve code quality and productivity.

CO 5	**Designing** and implement projects integrating multiple programming languages.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: GO programming Language**	**No. of hours: 10**
Content Summary: **Overview and Comparison:** Overview of GO, F#, Clojure, and Kotlin, Comparison with traditional programming languages, Installation and setup of development environment, **GO Programming Basics:** Introduction to GO syntax and data types, Control structures in GO, Functions and packages, Arrays, slices, and maps, Structs and custom data types,Pointers and memory management		
Unit Number: 2	**Title: F# Programming Language**	**No. of hours: 10**
Content Summary: Introduction to F# syntax and functional programming concepts, Data Types, Variables, Operators, Decision Making, Loops, Functions, Strings, Options, Immutable data types and pattern matching, Higher-order functions and currying, Asynchronous and parallel programming in F#, Object-Oriented Programming with F#, Database access with F#, Querying and manipulating data using F#, Integration with relational and NoSQL databases		
Unit Number: 3	**Title: Introduction to Clojure Programming**	**No. of hours: 10**
Content Summary: **Introduction to Clojure:** Overview of Clojure and its features, Setting up the development environment,		

Basic Syntax and Functional Programming, Basic syntax and data structures, Functional programming concepts, Immutable data and pure functions, Higher-order functions and recursion, Collections and sequence operations, Restructuring and pattern matching **Error Handling and Testing:** Exception handling and error management in Clojure, Testing strategies and frameworks in Clojure, **Data Manipulation and Transformation:** Data manipulation with Clojure's sequence functions, Data transformation with transducers, Data-driven development with data literals and data readers		
Unit Number: 4	**Title: Introduction to Kotlin Programming**	**No. of hours: 10**
Content Summary: Overview of Kotlin and its advantages, Setting up the development environment, Basic syntax and data types in Kotlin, Conditional statements and loops, Function declarations and parameters, Lambda expressions and higher-order functions, **Object-Oriented Programming in Kotlin**: Classes, objects, and inheritance, Properties and access modifiers, Interfaces and abstract classes, Understanding nullable and non-nullable types, Safe calls and the Elvis operator, Type inference and smart casting, **Collections and Functional Programming:** Working with lists, sets, and maps in Kotlin, Collection operations and transformations, Introduction to functional programming concepts in Kotlin, Creating extension functions in Kotlin, Using DSLs for domain-specific problems, Builder pattern and DSL implementation.		

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Explore new programming languages through engaging presentations.
2. **Problem-Based Assignments**: Analyze real-world scenarios to enhance programming skills.

3. **Project Labs**: Build and test applications using various new programming languages in hands-on labs.
4. **Collaborative Work**: Collaborate on case studies related to modern programming practices.
5. **Continuous Feedback**: Receive regular assessments and instructor feedback to monitor progress.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle for flexible learning.
2. **Question Bank**: Utilize a question bank and model papers for effective exam preparation.
3. **Real-World Applications**: Apply concepts from new programming languages to practical scenarios.

Text Books:

1. The Go Programming Language, Alan A. A. Donovan and Brian W. Kernighan, Addison-Wesley Professional.
2. An Introduction to Programming in Go, Caleb Doxsey, CreateSpace Independent Publishing.

References

1. Real-World Functional Programming: With Examples in F# and C#, Tomas Petricek and Jon Skeet, Manning.
2. Programming F# 3.0: A Comprehensive Guide for Writing Simple Code to Solve Complex Problems, Chris Smith, O'Reilly Media.
3. Getting Clojure: Build Your Functional Skills One Idea at a Time, Russ Olsen, O'Reilly.
4. The Joy of Clojure, Michael Fogus and Chris Houser, Manning Publication.
5. Atomic Kotlin, Bruce Eckel and Svetlana Isakova, Mindview LLC.
6. Kotlin in Action, Dmitry Jemerov and Svetlana Isakova, Manning Publication.

Additional Readings:

Online Learning Resources for New-Age Programming Languages

a) **Go (Golang)**

1. **Coursera: Programming with Google Go**
    1. **Description:** An introductory course to Go programming, covering language syntax, data structures, and more.
    2. **Link:** Coursera - Programming with Google Go
2. **Go by Example**
    1. **Description:** A hands-on introduction to Go using annotated example programs.
    2. **Link:** Go by Example

b) **F#**

1. **Microsoft Learn: Introduction to F#**
    1. **Description:** A series of modules introducing the F# language, its syntax, and functional programming concepts.
    2. **Link:** Microsoft Learn - Introduction to F#

c) **Clojure**

1. **ClojureBridge**
    1. **Description:** Free Clojure workshops for beginners, including resources and exercises.
    2. **Link:** ClojureBridge
2. **Learn Clojure: Clojure for the Brave and True**
    1. **Description:** A beginner-friendly book that teaches Clojure through real-world projects and examples.
    2. **Link:** Clojure for the Brave and True

d) **Kotlin**

1. **Kotlin Lang: Kotlin Documentation**
    1. **Description:** Official Kotlin documentation and tutorials by JetBrains.
    2. **Link:** Kotlin Documentation
2. **Udacity: Kotlin for Android Developers**
    1. **Description:** A course by Udacity focusing on Kotlin for Android development.
    2. **Link:** Udacity - Kotlin for Android Developers

NEW AGE PROGRAMMING LANGUAGES LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **New Age Programming languages Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP465	0-0-2	1
Type of Course:	Minor (Department Elective III)		
Pre-requisite(s), if any: Nil			

Course Outcomes (CO)

COs	Statements
CO1	**Understanding** the fundamental principles and paradigms of modern programming languages.
CO2	**Developing** proficiency in using the syntax, data structures, and control flow constructs of each language.
CO3	**Exploring** the unique features and strengths of each language, such as Go's focus on concurrency, F#'s functional programming capabilities, Clojure's emphasis on immutability and simplicity, and Kotlin's interoperability with existing Java code.
CO4	**Applying** the languages' respective development tools and best practices.
CO5	**Implementing** projects that utilize the strengths of each language to tackle complex problems or tasks.

Lab Experiments

S.N	Experiment Title	CO
1	Develop a RESTful API for a simple blog application in Go. The API should allow users to create, read, update, and delete blog posts. Use Go's built- in net/http package and struct types for handling blog post data	CO1
2	Create a command-line tool in Go that fetches and displays current weather information for a specified city. Use a public weather API and Go's JSON parsing capabilities to implement this tool.	CO1
3	Set up the F# development environment. Create a simple F# program to demonstrate basic syntax, data types, and variables.	CO1
4	Develop a functional calculator application in F#. The calculator should support basic arithmetic operations, as well as more advanced functions like trigonometry and logarithms. Use pattern matching and immutable data structures to handle calculations.	CO2
5	Create a small web application in F# using Suave (a lightweight web server library). The application should allow users to register, log in, and create simple posts. Implement basic session management and data storage.	CO2
6	Build a financial portfolio tracker in F#. The application should allow users to input and track their investments, calculate current value, and generate reports. Use F#'s asynchronous programming capabilities to fetch real-time stock prices from a financial API.	CO2
7	Develop a to-do list application in Clojure. The application should allow users to add, remove, and mark tasks as complete. Use Clojure's sequence operations and immutable data structures to	CO3

	manage tasks.	
8	Create a simple web scraper in Clojure. The scraper should fetch data from a specified website, parse the HTML content, and extract specific information. Use Clojure's libraries for HTTP requests and HTML parsing.	CO3
9	Develop a Kotlin-based Android application for tracking fitness activities. The app should allow users to log their workouts, view statistics, and set goals. Use Kotlin's object-oriented features and Android SDK for development.	CO4
10	Create a Kotlin DSL (Domain-Specific Language) for generating HTML pages. The DSL should allow users to define HTML structures using Kotlin syntax and generate the corresponding HTML code.	CO4

Summer Internship-III

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Summer Internship-III	**Course Code**	**L-T-P**	**Credits**
	ENSI451	---	2
Type of Course:	INT		
Pre-requisite(s), if any: NA			

Course Outcomes (CO)

CO1	Applying theoretical knowledge from core subjects to real-world problems in an industry or academic setting.
CO2	Demonstrating the acquisition of new technical skills relevant to the fieldof computer science and engineering during the internship.
CO3	Developing a comprehensive case study, project, or research paper that reflects the practical application of internship experiences.
CO4	Presenting internship outcomes effectively, showcasing professional growth, technical competencies, and communication skills.

Duration:

The internship will last for six weeks. It will take place after the completion of the 6th semester and before the commencement of the 7th semester.

Internship Options:

Students can choose from the following options:

- **Industry Internship (Offline) or Internship in Renowned Academic Institutions (Offline):**
  - Students must produce a joining letter at the start and a relieving letter upon completion.

Report Submission and Evaluation:

- Students must prepare a detailed report documenting their internship experience and submit it to the department. A copy of the report will be kept for departmental records.

2. **Case Study/Project/Research Paper:**
   - Each student must complete one of the following as part of their internship outcome:
     1. A case study
     2. A project
     3. A research paper suitable for publication
3. **Presentation:**
   - Students are required to present their learning outcomes and results from their summer internship as part of the evaluation process.

Evaluation Criteria for Summer Internship (Out of 100 Marks):

valuation Criteria	**Maximum Marks**
Relevance to Learning Outcomes	**30 Marks**
- Case Study/Project/Research Paper Relevance	15 Marks
- Application of Theoretical Knowledge	15 Marks
Skill Acquisition	**40 Marks**
- New Technical Skills Acquired	20 Marks
- Professional and Soft Skills Development	20 Marks
Report Quality	**15 Marks**
- Structure and Organization	8 Marks
- Clarity and Comprehensiveness	7 Marks

Presentation	**15 Marks**
- Content Delivery	8 Marks
- Visual Aids and Communication Skills	7 Marks

| **Total** | **100 Marks** |

Detailed View:

1. **Relevance to Learning Outcomes (30 Marks)**
   - **Case Study/Project/Research Paper Relevance (15 Marks):**
     1. Directly relates to core subjects: 15 marks
     2. Partially relates to core subjects: 10 marks
     3. Minimally relates to core subjects: 5 marks
     4. Not relevant: 0 marks
   - **Application of Theoretical Knowledge (15 Marks):**
     1. Extensive application of theoretical knowledge: 15 marks
     2. Moderate application of theoretical knowledge: 10 marks
     3. Minimal application of theoretical knowledge: 5 marks
     4. No application of theoretical knowledge: 0 marks
2. **Skill Acquisition (40 Marks)**
   - **New Technical Skills Acquired (20 Marks):**
     1. Highly relevant and advanced technical skills: 20 marks
     2. Moderately relevant technical skills: 15 marks
     3. Basic technical skills: 10 marks
     4. No new skills acquired: 0 marks
   - **Professional and Soft Skills Development (20 Marks):**
     1. Significant improvement in professional and soft skills: 20 marks
     2. Moderate improvement in professional and soft skills: 15 marks
     3. Basic improvement in professional and soft skills: 10 marks
     4. No improvement: 0 marks
3. **Report Quality (15 Marks)**
   - **Structure and Organization (8 Marks):**

1. Well-structured and organized report: 8 marks
2. Moderately structured report: 6 marks
3. Poorly structured report: 3 marks
4. No structure: 0 marks

- **Clarity and Comprehensiveness (7 Marks):**

Clear and comprehensive report: 7 marks

1. Moderately clear and comprehensive report: 5 marks
2. Vague and incomplete report: 2 marks
3. Incomprehensible report: 0 marks

4. **Presentation (15 Marks)**
   - **Content Delivery (8 Marks):**
     1. Clear, engaging, and thorough delivery: 8 marks
     2. Clear but less engaging delivery: 6 marks
     3. Somewhat clear and engaging delivery: 3 marks
     4. Unclear and disengaging delivery: 0 marks
   - **Visual Aids and Communication Skills (7 Marks):**
     1. Effective use of visual aids and excellent communication skills: 7 marks
     2. Moderate use of visual aids and good communication skills: 5 marks
     3. Basic use of visual aids and fair communication skills: 2 marks
     4. No use of visual aids and poor communication skills: 0 marks

Total: 100 Marks

K.R. MANGALAM UNIVERSITY

Career Readiness Boot Camp

Program	**Bachelor in Computer Applications (BCA)**		
Course Name: **Career Readiness Boot Camp**	Course Code	L-T-P	Credits
		2-0-0	2
Type of Course:	SEC		
Pre-requisite(s), if any: Basics of Java/C++ Programming, DBMS, Data Structure			

Preface:

This comprehensive course is meticulously designed to equip students with essential skills in Data Structures, Algorithms, Object-Oriented Programming, Java, and Database Management Systems, along with essential soft skills and aptitude preparation. Through a combination of self-paced online modules, hybrid learning experiences, and offline assessments, students will gain a profound understanding of foundational and advanced concepts crucial for software development and data management. The course places a strong emphasis on practical applications, problem-solving techniques, and the development of robust, maintainable code.

In addition to the core technical skills, students will be prepared for real-world scenarios through aptitude exams, soft skills training, and mock interviews, ensuring they are well-rounded and industry-ready. A key feature of this course is the integration of free certifications from Infosys Springboard, providing students with recognized credentials that enhance their employability and align with industry standards.

To successfully complete the course, students are required to pass each individual component, contributing to the final mark out of 100. This course is an integral part of the Practical Training Module (Bootcamp training) in the curriculum, and upon successful completion, students will earn 2 credits. The rigorous evaluation process ensures that students not only acquire the necessary knowledge but also demonstrate their ability to apply these skills in professional settings.

Students must obtain specific free certifications from Infosys Springboard (https://infytq.onwingspan.com/web/en/page/home).

Course Outcomes (COs):

CO1: Demonstrate understanding of basic data structures (arrays, strings, linked lists) and their operations to solve simple computational problems

CO2: Apply advanced data structures (stacks, queues, trees, graphs) to develop efficient algorithms for complex problem-solving.

CO3: Design and implement software solutions using Object-Oriented Programming principles, ensuring code reusability and maintainability.

CO4: Develop robust Java applications, utilizing object-oriented features, exception handling, and file I/O operations effectively

CO5: Analyze and optimize database systems using SQL for efficient data retrieval, transaction management, and ensuring data integrity.

SESSION WISE DETAILS

Module: 1	Data Structures and Algorithms - Part 1	No. of hours: 30 (Online, Self-Paced)
Content Summary: Module 1 covers foundational data structures including arrays, strings, and linked lists. It introduces key operations and practical applications. This foundational knowledge is crucial for advancing to more complex data structures.		
Module: 2	Data Structures and Algorithms - Part 2	No. of hours: 30 (Online, Self-Paced)
Content Summary: Module 2 focuses on advanced data structures such as stacks, queues, trees, and graphs. It covers essential operations, real-world applications, and their role in solving complex problems. Understanding these structures is vital for effective problem-solving and algorithm optimization.		
Module: 3	Object Oriented Programming	No. of hours:45 (Online, Self-Paced)
Content Summary: covers the fundamental concepts of OOP, including classes, objects, inheritance, polymorphism, and encapsulation. It emphasizes designing and implementing software using these principles to enhance code reusability and maintainability.		
Module: 4	Programming using Java	No. of hours: 100 (Online, Self-Paced)

Content Summary: basics of Java, including syntax, data types, operators, and control structures. It covers object-oriented principles specific to Java, such as classes, objects, inheritance, and polymorphism, along with advanced topics like exception handling and file I/O. The module aims to build strong foundational skills in Java for developing robust applications.		
Module: 5	Database management Systems (part I)	No. of hours: 60 (Online, Self-Paced)
Content Summary: fundamental concepts of database systems, including database models, relational databases, and SQL. It covers key topics such as entity-relationship modeling, normalization, and basic query operations. The module provides a solid foundation in designing and managing databases, focusing on effective data retrieval and manipulation using SQL.		
Module: 6	Database management Systems (part II)	No. of hours: 40 (Online, Self-Paced)
Content Summary: advanced database concepts, including transaction management, concurrency control, and database security. This module covers complex SQL queries, stored procedures, and triggers, as well as performance optimization techniques. It focuses on enhancing your skills in managing and optimizing large-scale databases, ensuring data integrity, and implementing robust security measures.		
Module: 7	Aptitude Exam	No. of hours: Online
Module: 8	Independent Evaluation through 3rd party	No. of hours: Offline
Module: 9	Softs Skills	No. of hours: Online
Module: 10	MOCK Interview	No. of hours: Hybrid

Reference Books:

- "Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein
- "Cracking the Coding Interview" by Gayle Laakmann McDowell
- "Elements of Programming Interviews" by Adnan Aziz, Tsung-Hsien Lee, and Amit Prakash

- "Head First Java" by Kathy Sierra and Bert Bates
- "Database System Concepts" by Abraham Silberschatz, Henry F. Korth, and S. Sudarshan
- "Introduction to the Theory of Computation" by Michael Sipser
- "Programming Challenges: The Programming Contest Training Manual" by Steven S. Skiena and Miguel A. Revilla
- "The Algorithm Design Manual" by Steven S. Skiena
- "Algorithms" by Robert Sedgewick and Kevin Wayne
- "Effective Java" by Joshua Bloch

Evaluation Criteria:

Module	Link	Hrs	Marks
Data Structures and Algorithms - Part 1	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01254096991326208 01065_shared/overview	30	10
Data Structures and Algorithms - Part 2	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01276673846938828 83448_shared/overview	30	10
Object Oriented Programming	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01254097227492556 81063_shared/overview	40	10
Programming using Java	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01288046454761881 6347_shared/overview	100	10
Database management Systems (part I)	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01275806667282022 456_shared/overview	60	10
Database management Systems (Part II)	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01276730056291942 41_shared/overview	40	
Aptitude Exam	Online Sessions to be conducted by	NA	10

	KRMU		
AMCAT (Aspiring Minds Computer Adaptive Test)	To be organized by KRMU through External agency	NA	20
Softs Skills	Online Sessions to be conducted by KRMU	NA	10
MOCK Interview	To be organized by KRMU	NA	10
			100

*Please Note: No end term evaluations will be done

Student Learning Experiences

- Foundational Mastery: Students will gain a strong foundation in data structures and algorithms, essential for advanced problem-solving in computer science.
- Hands-On Programming: Through extensive Java programming exercises, students will develop robust applications, reinforcing object-oriented principles.
- Advanced Database Skills: The course will enhance students' ability to manage and optimize databases, ensuring data integrity and effective transaction management.
- Practical Problem Solving: Students will apply advanced data structures to solve complex problems, preparing them for real-world challenges.
- Real-World Readiness: Soft skills training and mock interviews will equip students with the interpersonal and professional skills needed for career success.
- Self-Paced Learning: The online, self-paced modules allow students to learn at their own pace, ensuring a deep and thorough understanding of each topic.
- Comprehensive Evaluation: Independent evaluations and aptitude exams will rigorously assess students' knowledge and readiness for the tech industry.
- Certification and Credibility: Earning certifications from Infosys Springboard will enhance students' resumes and demonstrate their competence in industry-relevant skills.

(DEPARTMENT ELECTIVE-IV)

SECURE CODING AND VULNERABILITIES

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Secure Coding & Vulnerabilities**	**Course Code**	**L-T-P**	**Credits**
	ENSP301	4-0-0	4
Type of Course:	Minor (DEPARTMENT ELECTIVE-IV)		
Pre-requisite(s), if any:			

Course Perspective. This course provides an in-depth exploration of secure coding practices and the identification and mitigation of common vulnerabilities in software development. Students will gain a solid foundation in security concepts, secure application design, and the implementation of security best practices throughout the software development lifecycle. By understanding the principles of secure coding and the types of vulnerabilities that can compromise applications, students will be equipped to develop robust, secure software. The course covers essential topics such as input validation, authentication, cryptography, buffer overflows, SQL injection, and application security testing. The course is divided into four modules:

a) Introduction to Coding and Security
b) Secure Application Design and Architecture
c) Secure Coding Practices and Vulnerabilities
d) Application Security Testing and Deployment

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understand** different types of application security threats and their potential impact.
CO 2	**Apply** secure design principles and architectures to develop robust and secure applications.
CO 3	**Implement** secure coding practices for input validation, authentication, cryptography, session management, and error handling.
CO 4	**Conduct** static and dynamic application security testing to identify vulnerabilities and implement secure deployment and maintenance practices.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Introduction to coding and Security**	**No. of hours: 10**
Content Summary: Introduction-security concepts-CIA Triad, Viruses, Trojans, and Worms, threat, vulnerability, risk, attack. Coding Standards: Dirty Code and Dirty Compiler, Dynamic Memory Management functions, Common memory management Errors (Initialization Errors, Forget to Check Return Values, accessing already freed memory, Freeing the same memory multiple times, Forget to free the allocated memory), Integer Security –Introduction to integer types: Integer Data Types, data type conversions, Integer vulnerabilities and mitigation strategies		
Unit	**Title: Secure Application Design and**	**No. of hours: 10**

Number: 2	Architecture	
Content Summary: Security requirements gathering and analysis, Secure software development life cycle (SSDLC), Security issues while writing SRS, Design phase security, Development Phase, Test Phase, Maintenance Phase, Writing Secure Code – Best Practices SD3 (Secure by design, default and deployment), Security principles and Secure Product Development Timeline.		
Unit Number: 3	**Title: Secure Coding Practices and Vulnerabilities**	**No. of hours: 10**
Content Summary: Input validation Techniques-whitelist validation, regular expressions, authentication and authorization, Cryptography, buffer overflows, Session management and protection against session-related attacks, Secure error handling and logging practices, SQL Injection Techniques and Remedies, Race conditions		
Unit Number: 4	**Title: Application Security Testing and Deployment**	**No. of hours: 10**
Content Summary: Security code overview, Secure software installation. The Role of the Security Tester, Building the Security Test Plan. Testing HTTP-Based Applications, Testing File-Based Applications, Testing Clients with Rogue Servers, Static and Dynamic Application Security Testing (SAST & DAST), Secure Deployment and Maintenance, Patch management and software updates, Vulnerability scanning and penetration testing.		

Learning Experiences:

- Hands-on Vulnerability Testing: Practice identifying and mitigating common software vulnerabilities.
- Code Review Sessions: Conduct peer code reviews to identify potential security flaws and improve secure coding practices.
- Case Studies: Analyze real-world security breaches to understand how vulnerabilities are exploited.

- Interactive Labs: Implement secure coding techniques like input validation and buffer overflow protection.
- Security Audits: Perform security audits on sample applications to assess their vulnerability to SQL injection, session hijacking, and more.
- Role Play: Simulate the role of both attacker and defender in vulnerability exploitation and mitigation scenarios.
- Project-Based Learning: Develop secure applications using best practices in secure design, coding, and testing.
- Tools Exploration: Learn to use static and dynamic application security testing tools (SAST & DAST) for real-world applications.
- Collaborative Learning: Work in groups to design security testing plans and assess security risks in various application environments.
- Real-World Simulations: Conduct vulnerability scanning and penetration testing in simulated deployment environments.

Text Books and References

1. Secure Coding: Principles and Practices, Mark G. Graff, Kenneth R. Van Wyk, O'Reilly Media
2. Writing Secure Code, Michael Howard and David LeBlanc,Microsoft Press, 2nd Edition, 2004
3. Buffer Overflow Attacks: Detect, Exploit, Prevent by Jason Deckard ,Syngress,1st Edition, 2005
4. Threat Modeling, Frank Swiderski and Window Snyder,Microsoft Professional, 1st Edition ,2004
5. Secure Coding: Principles and Practices by Mark G. Graff, Kenneth R. van Wyk, Publisher(s): O'Reilly Media, Inc., 2003
6. The Software Vulnerability Guide (Programming Series) by H. Thompson (Author), Scott G. Chase, 2005

Additional Readings:

Online Learning References for "Secure Coding and Vulnerabilities"

1. **OWASP - Secure Coding Practices - Quick Reference Guide**
    - This guide provides a quick reference to secure coding practices based on OWASP's recommendations for secure software development.
    - Link: OWASP - Secure Coding Practices
2. **NPTEL - Secure Coding**
    - Offered by IITs through NPTEL, this course covers secure coding practices and principles for writing secure software.
    - Link: NPTEL - Secure Coding
3. **Mozilla Developer Network (MDN) - Web Security**
    - Comprehensive documentation on web security principles, secure coding practices, and common vulnerabilities in web applications.
    - Link: MDN - Web Security
4. **Google Code University - Web Security**
    - Learn about web security from Google, including secure coding practices and how to protect web applications from common threats.
    - Link: Google Code University - Web Security

SECURE CODING AND VULNERABILITIES LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Secure Coding & Vulnerabilities Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP351	0-0-2	1
Type of Course:	Minor (DEPARTMENT ELECTIVE-IV)		
Pre-requisite(s), if any:			

Lab Experiments

Defined Course Outcomes

COs	
CO 1	**Implement** fundamental security concepts such as the CIA Triad (Confidentiality, Integrity, Availability) and demonstrate secure coding practices to prevent common vulnerabilities.
CO 2	**Analyze** and fix memory management errors and integer vulnerabilities, applying mitigation strategies to enhance software security.
CO 3	**Develop** secure software by following the Secure Software Development Life Cycle (SSDLC), incorporating security principles and best practices throughout the development process.
CO 4	**Design** and test secure applications, performing vulnerability scanning, penetration testing, and implementing security measures to protect against attacks such as SQL injection and buffer overflow.

Ex. No	Experiment Title	Mapped CO/COs
P1	**Project Title: Secure Memory Management System** **Problem Statement:** Develop a secure memory management system for a critical application such as a healthcare management system. This system should handle dynamic memory allocation and deallocation securely, preventing common memory management vulnerabilities.	CO1
P2	**Project Title: Secure E-commerce Platform Design** **Problem Statement:** Design and implement a secure e-commerce platform that ensures data security throughout the software development life cycle (SDLC). The platform should handle sensitive user information securely and provide a robust security architecture.	CO2
P3	**Project Title: Secure Banking Application** **Problem Statement:** Develop a secure online banking application that ensures the protection of user data and prevents common vulnerabilities such as SQL injection, buffer overflow, and session hijacking.	CO3
P4	**Project Title: Comprehensive Security Testing and Deployment for a Social Media Platform** **Problem Statement:** Develop a social media platform with a focus on security testing and secure deployment. The platform should protect user data and provide a secure environment for social interactions.	CO4

CYBER CRIME INVESTIGATION & DIGITAL FORENSICS

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Cyber Crime Investigation & Digital Forensics**	**Course Code**	**L-T-P**	**Credits**
	ENSP303	4-0-0	4
Type of Course:	Minor (DEPARTMENT ELECTIVE-IV)		
Pre-requisite(s), if any:			

Course Perspective. The course offers an in-depth exploration of the methodologies and techniques employed in identifying, investigating, and prosecuting cybercrimes. As digital technologies permeate every aspect of modern life, understanding how to safeguard and investigate electronic evidence becomes crucial for ensuring security and justice. This course covers the foundational concepts of digital forensics, types of cybercrimes, investigation procedures, and the utilization of forensic tools. It prepares students to handle and analyze digital evidence proficiently, contributing to the effective enforcement of cyber laws. The course is divided into four comprehensive units:

a) Introduction
b) Types of Cyber Crimes
c) Investigation of Cyber Crimes
d) Forensic Tools and Processing of Electronic Evidence

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	Understand the nature and classification of conventional and cyber-crimes.
CO 2	Analyze and identify various types of cyber-crimes and their modes of operation.
CO 3	Evaluate the impact of cyber-crimes on individuals, organizations, and society.
CO 4	Develop an understanding of digital forensics and the investigative procedures used in cyber-crime cases.
CO 5	Apply forensic tools and techniques to retrieve and analyze digital evidence.

Course Outline:

Unit Number: 1	**Title: Title: Introduction**	**No. of hours: 10**
Content: Introduction to Digital Forensics, Definition and types of cybercrimes, electronic evidence and handling, electronic media, collection, searching and storage of electronic media, introduction to internet crimes, hacking and cracking, credit card and ATM frauds, web technology, cryptography, emerging digital crimes and modules.		
Unit Number: 2	**Title: Types of Cyber Crimes**	**No. of hours: 10**

Content: Crimes targeting Computers: Unauthorized Access Packet Sniffing Malicious Codes including Trojans, Viruses, Logic Bombs, etc. Online based Cyber Crimes: Phishing and its variants Web Spoofing and E-mail Spoofing Cyber Stalking Web defacement financial crimes, ATM and Card Crimes etc. Spamming Commercial espionage and Commercial Extortion online Software and Hardware Piracy Money Laundering Fraud& Cheating Other Cyber Crimes.		
Unit Number: 3	**Title: Investigation of Cyber Crimes**	**No. of hours: 10**
Content: Investigation of malicious applications Agencies for investigation in India, their powers and their constitution as per Indian Laws Procedures followed by First Responders; Evidence Collection and Seizure Procedures of Digital mediums Securing the Scene, Documenting the Scene, Evidence Collection and Transportation Data Acquisition Data Analysis Reporting		
Unit Number: 4	**Title: Forensic Tools and Processing of Electronic Evidence**	**No. of hours: 10**
Content: Introduction to Forensic Tools, Usage of Slack space, tools for Disk Imaging, Data Recovery, Vulnerability Assessment Tools, Encase and FTK tools, Anti Forensics and probable counters, retrieving information, process of computer forensics and digital investigations, processing of digital evidence, digital images, damaged SIM and data recovery, multimedia evidence, retrieving deleted data: desktops, laptops and mobiles, retrieving data from slack space, renamed file, ghosting, compressed files.		

Learning Experiences

- Interactive Lectures and Video Sessions:Students will engage with interactive PowerPoint presentations and video lectures for critical concepts in cyber crime and digital forensics, enhancing understanding through visual and auditory learning.

- Problem-Based Theory Assignments:Theory assignments will focus on real-world cyber crime scenarios, encouraging students to analyze and solve complex problems, thus bridging the gap between theory and practical application.
- Project-Based Lab Assignments:Hands-on lab projects will involve using forensic tools and techniques to investigate simulated cyber crimes, providing practical experience in evidence collection, data analysis, and the application of forensic methodologies.
- Collaborative Group Work:Students will work in groups to tackle case studies and forensic investigations, promoting teamwork, critical thinking, and peer learning. Group activities will include collaborative problem-solving and sharing insights on cyber security issues.
- Continuous Assessment and Feedback:Ongoing assessments, including quizzes, assignments, and practical labs, will monitor progress. Students will receive regular feedback from the course in charge, with opportunities for additional support and guidance as needed.
- Use of ICT Tools and Moodle LMS:All course materials, including lecture notes, assignments, and model question papers, will be accessible through Moodle LMS. Students will also use interactive teaching boards during lectures to visualize complex concepts and engage in real-time discussions.
- Engagement with a Question Bank and Model Papers:A comprehensive question bank and model question papers will be provided to aid in exam preparation, helping students familiarize themselves with potential exam questions and assess their understanding of the syllabus.
- Application of Theoretical Knowledge to Practical Scenarios:Students will apply theoretical knowledge to practical cases and simulations, enabling them to experience the full cycle of a cyber crime investigation, from evidence collection to analysis and reporting.

Text Books & References

1. Moore, Robert, (2011). Cybercrime, investigating high-technology computer crime(2nd Ed.). Elsevie

2. C. Altheide& H. Carvey Digital Forensics with Open Source Tools, Syngress, 2011.
3. Majid Yar, "Cybercrime and Society", SAGE Publications Ltd, Hardcover, 2nd Edition, 2013.
4. Robert M Slade, "Software Forensics: Collecting Evidence from the Scene of a Digital Crime", Tata McGraw Hill, Paperback, 1st Edition, 2004.

Additional Readings:

Online Learning References:

I) **Cybrary - Digital Forensics**
   a. A free online course that covers various aspects of digital forensics, including tools, techniques, and procedures for investigating cybercrimes.
   b. Link: Cybrary - Digital Forensics

II) **Pluralsight - Digital Forensics Fundamentals**
   a. This course offers a thorough understanding of digital forensics, covering the fundamentals, tools, and techniques used in the field.
   b. Link: Pluralsight - Digital Forensics Fundamentals

III) **SANS Institute - Digital Forensics and Incident Response Blog**
   a. A blog providing insights, case studies, and updates on the latest in digital forensics and incident response.
   b. Link: SANS Institute - Digital Forensics Blog

IV) **OWASP - Open Web Application Security Project**
   a. Provides resources on web security, including best practices for secure coding and tools for vulnerability assessment, which are essential for investigating cybercrimes.
   b. Link: OWASP - Open Web Application Security Project

CYBER CRIME INVESTIGATION & DIGITAL FORENSICS LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Cyber Crime Investigation & Digital Forensics Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP353	0-0-2	1
Type of Course:	Minor (DEPARTMENT ELECTIVE-IV)		
Pre-requisite(s), if any:			

Proposed Lab Experiments

Defined Course Outcomes

COs	
CO 1	**Understand** the fundamental concepts and principles of digital forensics and cybercrimes.
CO 2	**Apply** the knowledge of digital forensics techniques and procedures to collect, analyse, and preserve electronic evidence in various types of cybercrimes.
CO 3	**Evaluate** and utilize forensic tools and technologies for data acquisition, analysis, and recovery in the investigation of cybercrimes.
CO 4	**Analyse** and interpret digital evidence obtained from different sources, such as electronic media, internet crimes, malicious applications, and various forms of cybercrimes.

	Experiment Title	Mapped CO/COs
1	**Project Title: Comprehensive Study on Cybercrime and Digital Forensics** **Problem Statement:** Conduct a comprehensive study on various types of cybercrimes and the role of digital forensics in investigating these crimes. The project will involve collecting electronic evidence, understanding cybercrime techniques, and applying digital forensics methodologies.	CO1
2	**Project Title: Simulation and Prevention of Cyber Crimes** **Problem Statement:** Develop a comprehensive simulation and prevention strategy for various types of cybercrimes. The project will involve creating scenarios for unauthorized access, phishing, and malware attacks, and implementing preventive measures.	CO2
3	**Project Title: Investigation and Reporting of Cyber Crime Incidents** **Problem Statement:** Investigate a simulated cybercrime incident, collect and analyze digital evidence, and report the findings. The project will cover the entire investigation process from securing the scene to data analysis and reporting.	CO3
4	**Project Title: Advanced Digital Forensics and Evidence Processing** **Problem Statement:** Develop a system for advanced digital forensics and processing of electronic evidence. The project will involve using forensic tools for data recovery, vulnerability assessment, and processing digital evidence from various devices.	CO4

AI IN CYBER SECURITY

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **AI in Cyber Security**	**Course Code**	**L-T-P**	**Credits**
	ENSP305	4-0-0	4
Type of Course:	Minor (DEPARTMENT ELECTIVE-IV)		
Pre-requisite(s), if any:			

Course Perspective. The course delves into the integration of Artificial Intelligence (AI) techniques within the realm of cyber security, highlighting the transformative potential of AI in detecting, preventing, and responding to cyber threats. As cyber threats evolve in complexity and scale, AI offers advanced methodologies to enhance security measures and mitigate risks effectively. This course provides a comprehensive understanding of the applications of AI in cyber security, from fundamental machine learning and deep learning techniques to their practical implementations in threat detection and prevention.

Students will explore the history, evolution, and current trends of AI in cyber security, gaining insights into the ethical considerations and challenges associated with the adoption of AI technologies in this critical field. Through detailed case studies and practical examples, the course bridges theoretical concepts with real-world applications, equipping students with the skills necessary to leverage AI for robust cyber defense strategies.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understand** the concepts and applications of AI in the field of cyber security.
CO 2	**Express** the ethical and legal considerations associated with the use of AI in cyber security.
CO 3	**Determine** emerging trends and technologies in AI for cyber security, and their potential impact on the field.
CO 4	**Identify** strategies for integrating AI-driven solutions into existing cyber security frameworks, policies, and practices.
CO 5	**Articulate** critical thinking and problem-solving skills to address real-world cyber security challenges using AI techniques.
CO 6	**Design** machine learning techniques for threat detection and prevention in cyber security, including supervised and unsupervised algorithms.

Course Outline:

Unit Number: 1	**Title: Introduction to AI and Cyber Security**	**No. of hours: 10**
Content: Overview of Artificial Intelligence and its applications in Cyber Security Evolution and impact of AI on Cyber Security Understanding Cyber Security threats and the role of AI Basic principles of Machine Learning (ML) and Deep Learning (DL) in Cyber Security Ethical considerations and challenges of AI in Cyber Security		
Unit Number: 2	**Title: Machine Learning Techniques for Cyber Security**	**No. of hours: 10**

Content: Introduction to Machine Learning techniques relevant to Cyber Security Overview of supervised and unsupervised ML models Feature engineering and data preparation for ML models Practical applications and case studies of ML in Cyber Security.		
Unit Number: 3	**Title: Deep Learning Techniques for Cyber**	**No. of hours: 10**
Content: Introduction to Deep Learning and its significance in Cyber Security Applications of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) Overview of Generative Adversarial Networks (GANs) and their use in Cyber Security Case studies illustrating the use of DL techniques for Cyber Security problems		
Unit Number: 4	**Title: AI for Cyber Security: Threat Detection and Prevention**	**No. of hours: 10**
Content: AI applications in threat detection and prevention Overview of traditional vs. AI-driven threat detection methods Fundamentals of supervised and unsupervised ML algorithms for threat detection Advanced deep learning techniques for threat detection (CNNs, RNNs) Feature selection, emerging trends, and challenges in AI for Cyber Security		

Learning Experiences:

- Interactive Lectures: Engage with multimedia-rich lectures using PPTs and video materials, providing a thorough overview of AI applications in Cyber Security and foundational concepts of Machine Learning and Deep Learning.

- Hands-On Assignments: Apply theoretical knowledge through problem-based assignments and lab exercises, focusing on real-world scenarios of AI in Cyber Security, such as threat detection and prevention using ML and DL techniques.

- Case Studies and Discussions: Analyze and discuss case studies that demonstrate the practical application of ML and DL in Cyber Security, encouraging critical thinking and application of concepts to solve complex problems.

- Group Projects: Collaborate in groups to work on projects involving AI-driven Cyber Security solutions, fostering teamwork and peer learning, and providing practical experience in developing and implementing secure systems.

- Continuous Assessment: Participate in regular quizzes and assignments to assess understanding and application of course material, with opportunities for feedback and improvement based on performance.

- Use of ICT Tools: Utilize Moodle LMS for accessing course materials, submitting assignments, and engaging in interactive discussions, enhancing the learning experience with technology.

- Support and Feedback: Benefit from regular feedback and additional support from the course instructor, with opportunities to seek help and clarify doubts, ensuring a thorough understanding of complex topics.

Text Book References

1. Artificial Intelligence for Cybersecurity" by Bhaskar Sinha (Auerbach Publications)
2. Machine Learning and Security: Protecting Systems with Data and Algorithms" by Clarence Chio and David Freeman (O'Reilly Media)

Additional Readings:

Online Learning Resources:

I) **Cybrary - Introduction to Artificial Intelligence for Cyber Security**

a. This course offers insights into how AI can be applied to cyber security, including threat detection and response.

b. Link: Cybrary - Introduction to Artificial Intelligence for Cyber Security

II) **Pluralsight - Machine Learning and AI for Cybersecurity**

a. This course provides an in-depth look at how machine learning and AI can be used to enhance cyber security measures.

b. Link: Pluralsight - Machine Learning and AI for Cybersecurity

III) **FutureLearn - Artificial Intelligence for Cyber Security by Coventry University**

a. This course explores the application of AI in cyber security, covering topics like threat detection, response, and mitigation.

b. Link: FutureLearn - Artificial Intelligence for Cyber Security

IV) **MIT OpenCourseWare - Artificial Intelligence**

a. Lecture notes, assignments, and exams from MIT's course on Artificial Intelligence, providing a deep dive into AI concepts applicable to cyber security.

b. Link: MIT OpenCourseWare - Artificial Intelligence

V) **IBM - Introduction to Cyber Security Tools & Cyber Attacks**

a. A course that covers various cyber security tools and techniques, including the use of AI and machine learning for threat detection and prevention.

b. Link: IBM - Introduction to Cyber Security Tools & Cyber Attacks

AI IN CYBER SECURITY LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **AI in Cyber Security Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP355	0-0-2	1
Type of Course:	Minor (DEPARTMENT ELECTIVE-IV)		
Pre-requisite(s), if any: basic understanding of web development technologies such as HTML, CSS, and JavaScript. Additionally, students should have some familiarity with networking concepts, operating systems, and databases.			

Defined Course Outcomes

COs	Statement
CO 1	**Analyze** the history, evolution, and ethical considerations of AI in cyber security, documenting key milestones and advancements, and discussing the implications of AI applications.
CO 2	**Implement** and evaluate machine learning models for classifying and detecting cyber threats, using various datasets and techniques such as supervised and unsupervised learning, deep learning, and anomaly detection.
CO 3	**Develop** and apply feature engineering, data preparation, and model training techniques to enhance the performance and accuracy of cyber security models.
CO 4	**Conduct** comprehensive case studies and surveys on the application of AI in cyber security, identifying emerging trends, challenges, and documenting methodologies and findings.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
P1	**Project Title: Comprehensive Analysis of AI in Cyber Security** **Problem Statement:** Conduct a comprehensive analysis of the role of AI in cyber security. The project will involve studying the history, evolution, and current trends in AI applications for cyber security, and understanding the basic principles of machine learning and deep learning in this context.	CO1
P2	**Project Title: Machine Learning Models for Cyber Threat Detection** **Problem Statement:** Develop and evaluate different machine learning models for detecting cyber threats. The project will involve implementing supervised and unsupervised learning techniques, performing feature engineering, and analyzing case studies.	CO2
P3	**Project Title: Deep Learning Models for Advanced Cyber Threat Detection** **Problem Statement:** Develop and evaluate deep learning models for advanced cyber threat detection. The project will involve implementing CNNs, RNNs, and GANs, and analyzing their applications in cyber security.	CO3
P4	**Project Title: AI-Based Comprehensive Threat Detection System** **Problem Statement:** Develop a comprehensive AI-based system for threat detection and prevention in cyber security. The project will involve implementing machine learning and deep learning models and addressing the challenges of traditional threat detection methods.	CO4

SOCIAL MEDIA SECURITY

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: **Social Media Security**	**Course Code**	**L-T-P**	**Credits**
	ENSP307	4-0-0	4
Type of Course:	Minor (DEPARTMENT ELECTIVE-IV)		
Pre-requisite(s), if any:			

Course Perspective. This course introduces students to the critical concepts of social media security, addressing the growing need to understand and manage security and privacy issues in the digital age. Social media platforms have become integral to personal, professional, and commercial interactions, creating a complex landscape of potential security threats and privacy concerns. This course aims to equip students with the knowledge and skills required to navigate and mitigate these risks effectively. Students will explore the technical, legal, and social dimensions of social media security, developing strategies to safeguard personal information, ensure user trust, and comply with legal standards.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Demonstrate** an understanding of the different types of social media platforms, their features, and their impact on communication, marketing, and society.
CO 2	**Acquire** knowledge and skills in social media monitoring techniques, including data collection, analysis, and the use of relevant tools and technologies.
CO 3	**Develop** the ability to analyze and evaluate viral content on social media, understand the factors contributing to its spread, and recognize its implications for marketing and online engagement.
CO 4	**Identify** the challenges, opportunities, and pitfalls associated with social media marketing, and formulate strategies for effective audience targeting, engagement, and brand promotion.
CO 5	**Develop** strategies to safeguard personal information, foster user trust, and mitigate associated risks.

Course Outline:

Unit Number: 1	**Title: Social Media Overview**	**No. of hours: 10**
Content Summary: Overview of Social Media Types and Platforms Social Media Monitoring and Analysis Data Collection and Analysis Methods: BoW Model, TF-IDF		

Network Analysis Basics: Node Centrality, Degree Distribution, Clustering Coefficient Introduction to Synthetic Networks: Random Graphs, Preferential Attachment		
Unit Number: 2	**Title: S**ocial Media Management and Marketing	**No. of hours: 10**
Content Summary: Strategies for Recruitment and Employment Screening Customer Engagement and Content Management Evaluating Social Media Campaigns: Effective vs. Ineffective Ethical Considerations and Privacy in Crowdsourcing Managing and Promoting Social Media Presence		
Unit Number: 3	**Title: Privacy Issues in Social Media**	**No. of hours: 10**
Content Summary: Privacy Settings and Personal Identifiable Information (PII) Leakage Types of Privacy Attacks: Identity Disclosure, Inference Attacks, De-anonymization Privacy Metrics: k-anonymity, l-diversity, Differential Privacy Balancing Personalization and Privacy Impact of Privacy on User Trust		
Unit Number: 4	**Title: Social Media Security: Laws, Best Practices, and Case Studies**	**No. of hours: 10**
Content Summary: Legal Aspects: Posting and Content Regulations Best Practices: Content Moderation, User Authentication Security Awareness and Education Case Studies: Facebook, Twitter, Instagram, YouTube, LinkedIn, and others		

Learning Experiences

- Interactive Lectures and Discussions: Use Lecture PPTs and video lectures to introduce topics, followed by class discussions on current social media trends.

- Hands-on Data Analysis: Conduct practical sessions where students collect and analyze social media data using tools like BoW and TF-IDF.

- Group Social Media Campaign Projects: Students design, implement, and evaluate a mock social media campaign in teams, applying marketing strategies and content management principles.

- Case Study Analysis: Review and present case studies of social media platforms to understand real-world applications and issues.

- Privacy and Security Workshops: Use interactive tools and workshops to explore privacy settings, privacy metrics, and security measures.

- Ethical Dilemma Debates: Engage in debates on ethical issues and privacy concerns related to social media usage.

- Practical Management Tools Sessions: Train students on social media management tools and practices for content moderation and promotion.

- Continuous Assessment and Feedback: Implement regular quizzes, assignments, and peer reviews, with ongoing feedback to track progress.

- Moodle LMS and ICT Integration: Utilize Moodle LMS for course materials, assignments, and discussions, and encourage use of ICT tools for learning.

References

1. Mastering Social Media Mining, Bonzanini Marco, Packt Publishing Limited
2. Mining the Social Web, Mikhail Klassen and Matthew A. Russell, O'Reilly Media, Inc
3. Social media mining: an introduction, Zafarani, Reza, Mohammad Ali Abbasi, and Huan Liu, Cambridge University Press
4. Social Media Security: Leveraging Social Networking While Mitigating Risk, Michael Cross, Syngress
5. Social Media and the Law: A Guidebook for Communication Students and Professionals, Daxton R. Stewart, Taylor & Francis Ltd

6. Security in the Digital Age: Social Media Security Threats and Vulnerabilities by Henry A. Oliver, Create Space Independent Publishing Platform.

Additional Readings:

Online Learning Resources for Social Media Security

R 1. Coursera - Social Media Marketing Specialization

- **Provider:** Northwestern University
- **Description:** This specialization covers the major social media platforms, marketing strategies, and data analysis tools.
- **Link:** Coursera Social Media Marketing Specialization

R 2. edX - Cybersecurity Fundamentals

- **Provider:** Rochester Institute of Technology
- **Description:** This course offers foundational knowledge in cybersecurity, including threats, vulnerabilities, and defense strategies.
- **Link:** edX Cybersecurity Fundamentals

R 3. Udemy - The Complete Cyber Security Course: Network Security!

- **Instructor:** Nathan House
- **Description:** This comprehensive course covers network security, including how to secure your network, protect your devices, and more.
- **Link:** Udemy Cyber Security Course

SOCIAL MEDIA SECURITY LAB

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name: Social Media Security Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP357	0-0-2	1
Type of Course:	Minor (DEPARTMENT ELECTIVE-IV)		
Pre-requisite(s), if any:			

Course Outcomes (CO)

COs	Statements
CO1	**Analyze** different social media platforms, their features, and the ethical and privacy considerations in crowdsourcing and data handling.
CO2	**Implement** data collection, text analysis, and network analysis techniques on social media datasets, demonstrating proficiency in using APIs and various models.
CO3	**Develop** strategies and plans for using social media in various contexts such as employment screening, customer engagement, and small business promotion.
CO4	**Evaluate** privacy settings, metrics, and security incidents on social media platforms, applying best practices for user authentication, access control, and content moderation.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
P1	**Project Title: Comprehensive Analysis of Social Media Platforms** **Problem Statement:** Conduct a comprehensive analysis of different social media platforms, their features, and the data they generate. The project will involve collecting and analyzing data from social media, performing content analysis, and understanding network properties.	CO1
P2	**Project Title: Effective Social Media Management and Marketing Strategy** **Problem Statement:** Develop an effective social media management and marketing strategy for a small business. The project will involve analyzing customer engagement, creating marketing strategies, and addressing ethical considerations in social media use.	CO2
P3	**Project Title: Privacy Protection in Social Media** **Problem Statement:** Develop strategies and tools to protect user privacy on social media platforms. The project will involve analyzing privacy settings, simulating privacy attacks, and evaluating privacy metrics.	CO3
P4	**Project Title: Enhancing Security and Compliance on Social Media Platforms** **Problem Statement:** Develop a comprehensive approach to enhance security and ensure compliance with laws on social media platforms. The project will involve researching laws, developing best practices, and analyzing case studies of security incidents.	CO4

Semester: VIII

Industrial Project/R&D Project/Start-up Project

Program	BCA		
Course Name: Industrial Project/R&D Project/Start-up Project	Course Code	L-T-P	Credits
	ENSI452		12
Type of Course:	SEC		
Pre-requisite(s), if any:			

Preface:

The **BCA Semester Full-Time Project Work** is a culmination of the academic journey for engineering students at the School of Engineering & Technology, K.R. Mangalam University. This detailed Standard Operating Procedure (SOP) is designed to guide students through their project, ensuring a comprehensive, practical, and outcome-driven approach that aligns with the principles of the **National Education Policy (NEP) 2020**. The SOP provides a framework for students to choose from three types of projects—**Industrial Projects**, **Research & Development (R&D) Projects**, and **Start-up Projects**. It emphasizes experiential learning, real-world problem-solving, and interdisciplinary collaboration, reflecting NEP 2020's focus on holistic development, innovation, and entrepreneurship. Students will work under the mentorship of both internal faculty and external experts, ensuring they are equipped with the skills and knowledge required to excel in industry, research, or entrepreneurship.

This document outlines each stage of the project work, from proposal submission to final evaluation, and offers clear guidelines for successful completion. By adhering to this SOP, students will not only demonstrate their technical proficiency but also contribute meaningfully to industry, academia, and society.

Standard Operating Procedure (SOP) for BCA Final Semester Full-Time Project Work

1. Introduction

The **BCA Final Semester Full-Time Project Work** is an essential academic requirement aimed at providing students with the opportunity to apply theoretical knowledge to practical challenges. The project is designed to foster critical thinking, problem-solving, innovation, and research-oriented learning, with a focus on real-world industrial, research, and entrepreneurial domains. Students may choose from:

- **Industrial Project**: Solving real industrial problems in collaboration with an industry partner.
- **Research & Development (R&D) Project**: Contributing to academic and applied research, with external guidance from academic/research institutions.
- **Start-up Project**: Developing and launching innovative start-up ideas with entrepreneurial mentors.

The SOP ensures that the project aligns with **NEP 2020 guidelines**, emphasizing interdisciplinary, practical, and outcome-based learning.

---

2. Objectives

The primary objectives of the full-time project are:

- **Application of Theoretical Knowledge**: Enabling students to apply their academic learning to practical problems.
- **Holistic Development**: Promoting interdisciplinary learning, critical thinking, creativity, and problem-solving.
- **Research and Innovation**: Encouraging innovative solutions, leading to publications, patents, or prototypes.
- **Industry Collaboration**: Fostering partnerships with industries for real-world problem-solving.
- **Entrepreneurship Development**: Developing entrepreneurial skills and creating viable start-ups.

- **Global Competency**: Ensuring students develop the skills required to excel in global environments through research, innovation, and collaboration.

---

3. Types of Projects

a) Industrial Project

Students working on **Industrial Projects** will:

- Collaborate with an industry partner.
- Identify specific, real-world challenges faced by the company.
- Propose and implement a solution that provides value to the industry.
- Develop a final product or prototype that can be implemented in the industrial setting.

Project Proposal:

- Problem Statement and Objectives: Identify the industrial problem and outline the objectives.
- Proposed Solution: Present a detailed methodology for solving the problem.
- Deliverables: Define tangible deliverables, including prototypes, software, or hardware.
- Expected Impact: Outline the expected impact on the industry.

Evaluation Criteria:

- Practical implementation and solution viability (40%)
- Project innovation (20%)
- Industrial applicability and impact (20%)
- Final presentation and report quality (20%)

b) Research & Development (R&D) Project

The **R&D Project** focuses on creating innovative research outcomes through collaborations with academic or research institutions. This can result in publications, research reports, or new discoveries.

Project Proposal:

- Literature Review: Detailed research on existing work related to the chosen topic.
- Hypothesis/Research Questions: Define the specific research problem or question.
- Methodology: Include data collection, experimental design, and analysis techniques.
- Research Timeline: Step-by-step phases of research with milestones.

External Mentor: Collaboration with an **external academic expert** is mandatory for research projects. The external mentor must be a research professional with expertise in the specific field of study.

Internal Mentor: Each student will also be assigned an **internal faculty member** who will supervise the project. The internal mentor will ensure that the research meets academic standards and deadlines.

Evaluation Criteria:

- Quality of Research and Novelty (30%)
- Research Methodology (25%)
- Contributions to the field (20%)
- Final Report, Presentation, and Publication (25%)

c) Start-up Project

The **Start-up Project** involves developing a business model or creating a start-up venture. Students work on a product/service idea that addresses a significant market need or societal problem.

Project Proposal:

- Start-up Idea: Explain the business or product idea.
- Market Research: Detailed research on the market, target customers, competitors, and potential revenue streams.
- Business Plan: Define the steps needed to take the idea to market, including funding, development phases, marketing, and operational plans.
- Product Prototype: If applicable, develop a working prototype.

Mentorship:

- **External Mentor**: An industry/start-up expert will guide the student in refining the idea, business model, and market strategy.
- **Internal Faculty Mentor**: An internal mentor will provide academic guidance and ensure the start-up idea is feasible and innovative.

Evaluation Criteria:

- Start-up viability and market potential (30%)
- Product or service innovation (30%)
- Prototype/Business Model Development (20%)
- Final Pitch/Presentation and Start-up Plan (20%)

---

4. Roles and Responsibilities

a) Student's Responsibilities:

- Select a suitable project topic based on interests (industrial, R&D, or start-up).
- Draft and submit a detailed proposal with objectives, methodology, timelines, and deliverables.
- Coordinate with both external and internal mentors regularly for feedback and guidance.
- Maintain a weekly progress report for both mentors.
- Submit a final comprehensive report and present the project.

b) Internal Supervisor:

- Guide the student throughout the project.
- Provide academic input and ensure that the project aligns with the program outcomes.
- Conduct progress reviews and ensure timelines are adhered to.
- Evaluate the project at the mid-term and final stages.

c) External Mentor:

- Offer specialized industrial, research, or entrepreneurial guidance.

- Provide real-world problem insights for industrial and start-up projects.
- Ensure the project is relevant to the chosen industry, research domain, or start-up ecosystem.
- Participate in the final evaluation of the project.

---

5. Project Phases

Phase 1: Proposal Submission and Approval

- Students will submit a project proposal during the first two weeks of the final semester.
- The proposal must include the problem statement, objectives, literature review (for R&D projects), methodology, and expected outcomes.
- The proposal is subject to review and approval by the internal supervisor and external mentor.

Phase 2: Planning and Resource Allocation

- Once approved, the student will develop a project plan that includes:
    - **Project Milestones**: Break down the project into smaller tasks with defined milestones.
    - **Resource Requirements**: Identify any software, hardware, lab resources, or tools required for the project.
    - **Team Roles**: For group projects, define the roles of each team member.
    - **Risk Assessment**: Highlight potential risks and the corresponding mitigation strategies.

Phase 3: Mid-term Review

- A mid-term review will be conducted halfway through the project to assess progress.
- Students will present their work to a committee consisting of the internal supervisor, external mentor, and department head.
- The review will assess the progress against the timeline and suggest course corrections if needed.

Phase 4: Final Execution and Evaluation

- **Industrial Projects**: Students must submit a prototype or industrial report, demonstrating the solution's applicability to the industry.
- **R&D Projects**: Students must submit a final research report or publish findings in academic journals.
- **Start-up Projects**: Students must present a business plan, along with a working prototype, market analysis, and revenue model.

Phase 5: Final Report Submission and Presentation

- **Final Report**: The project report should contain a title page, abstract, introduction, problem statement, objectives, methodology, results, discussion, conclusions, future scope, references, and appendices.
- **Presentation**: Students will deliver a final presentation to a panel of evaluators, showcasing their work, findings, or product.
- **Evaluation**: Based on the final report and presentation, students will be awarded marks in accordance with the evaluation rubrics.

---

6. Collaboration and Mentorship

For **Research Projects**, the mentorship will involve both:

- **External Mentor**: An academic expert outside the institution, preferably from a reputed university or research institute.
- **Internal Mentor**: A faculty member from the student's department to provide academic and administrative guidance.

For **Industrial Projects**:

- External mentorship will come from industry professionals, preferably from the partnering company.

For **Start-up Projects**:

- External mentorship will involve experienced entrepreneurs, start-up founders, or investors.

Mentors will:

- Provide critical inputs on the technical, business, or research aspects of the project.
- Offer feedback and advice during each phase of the project.

---

7. NEP 2020 Guidelines

The project structure is designed to ensure interdisciplinary learning and foster entrepreneurial and research innovation, in line with the **NEP 2020** guidelines:

- **Interdisciplinary Approach**: Students are encouraged to explore projects that bridge different fields of study.
- **Flexibility**: Students have the flexibility to choose between industrial, research, or start-up projects.
- **Experiential Learning**: Real-world problem-solving and hands-on project work are at the core of this initiative.
- **Collaboration**: The integration of external mentors ensures industry and academic collaboration.

---

8. Documentation and Submission Requirements

Students are required to:

- Submit their proposal, mid-term report, final report, and any supporting documents via the **Learning Management System (LMS)**.
- Maintain detailed project logs and weekly reports.