



SNDT Women's University, Mumbai

Bachelor of Computer Applications (BCA)

Syllabus

as per NEP 2020

w.e.f.

A.Y.: 2024-2025.

1, Nathibai Thackersey Road, Mumbai- 400020

www.sndt.ac.in

Tentative Template

Abbreviation	Full-form	Remarks	Related to Major and Minor Courses
Major(Core)	Main Discipline		
Major(Elective)	Elective Options		related to the Major Discipline
Minor Stream	Other Disciplines (Inter/Multidisciplinary) not related to the Major	either from the same Faculty or any other faculty	
OEC	Open Elective Courses/Generic		Not Related to The Major and Minor
VSEC	Vocational and Skill Enhancement Courses		
VSC	Vocational Skill Courses		Not Related to The Major and Minor
SEC	Skill Enhancement Courses		Not Related to The Major and Minor
AEC	Ability Enhancement Courses	Communication skills, critical reading, academic writing, etc.	Not Related to the Major and Minor
VEC	Value Education Courses	Understanding India, Environmental science/education, Digital and technological solutions, Health & Wellness, Yoga education, sports, and fitness	Not Related to the Major and Minor

IKS	Indian Knowledge System	I. Generic IKS Course: basic knowledge of the IKS II. Subject Specific IKS Courses: advanced information pertaining to the subject: part of the major credit.	Subject Specific IKS related to Major
OJT	On-Job Training(Internship/Apprenticeship)	Corresponding to the Major Subject	Related to the Major
FP	Field projects	Corresponding to the Major Subject	Related to the Major
CC	Co-curricular Courses	Health and Wellness, Yoga education sports, and fitness, Cultural Activities, NSS/NCC and Fine/Applied/Visual/Performing Arts	Not Related to the Major and Minor
CE	Community Engagement and service		Not Related to the Major and Minor
RP	Research Project	Corresponding to the Major Subject	Related to the Major

Programme Template:

Programme Degree	Bachelor of Computer Applications (BCA)
Parenthesis	NA
Preamble(Brief Introduction to the programme)	<p>The Bachelor of Computer Applications (BCA) program is a four-year undergraduate degree program as per NEP-2020 designed to provide students with a strong foundation in computer science and its applications. The program aims to equip students with the knowledge and skills required to excel in the rapidly evolving field of computer science and information technology.</p> <p>The BCA program combines theoretical knowledge with practical applications to ensure that students develop a comprehensive understanding of computer systems, software development, database management, networking, and other core areas of computer science. It is an ideal choice for students who are interested in pursuing a career in the IT industry or furthering their studies in computer science.</p> <p>During the course of the BCA program, students are exposed to a wide range of subjects that cover various aspects of computer science. These subjects typically include programming languages, data structures, algorithms, computer architecture, operating systems, software engineering, web development, database management systems, computer networks, and information security.</p> <p>Upon successful completion of the BCA program, graduates have a wide range of career opportunities in the IT industry. They can work as software developers, system analysts, database administrators, network administrators, web developers, IT consultants, and other related roles. Graduates may also choose to pursue higher education, such as a Master's degree in computer science or a specialized field within the IT domain.</p> <p>By combining theoretical knowledge, practical skills, and industry exposure, the program equips students with the necessary tools to thrive in the IT industry and contribute to technological advancements.</p>
Programme Specific Outcomes(PSOs)	<p>After completing this programme, Learner will</p> <ol style="list-style-type: none"> 1. Build a strong foundation in computer application, including knowledge of Programming languages, Database, Mathematics, Operating system and Networking. 2. Understand the ethical and professional responsibilities in the field of computer applications by adhering to professional standards and practices. 3. Applying programming knowledge to develop a software application to solve specific problems.

	4.	Analyzing system requirements to design efficient and effective software solutions.
	5.	Evaluate software designs and architectures for efficiency, security and user experience.
	6.	Create a software application to meet the requirements of the Industrial Standards.
Eligibility Criteria for Programme		Eligibility: As Per AICTE and MAH-CET Cell norms. MAH-CET-BCA/MCA (Integrated)- CET Score
Intake (For SNTWU Departments and Conducted Colleges)		60

- *External Examination does not always mean Theory paper. It may practical examination, Product submission, projects, etc. checked by external examiners.*
- *Internal evaluation should not be written Theory papers like Unit tests. Internal marks will be acquired through practical, small group or individual Projects, activities, presentations, seminars, workshops, products, assignments, application-based work, reports, etc.*
- *Practical may be part of the main courses along with theory modules instead of having separate courses of practical work.*

Structure with Course Titles*(Options related to our area of study to be provided with "OR" for baskets of different types)*

SN	Courses	Type of Course	Credits	Marks	Int	Ext
Semester I						
1.1	Problem Solving using C	Major(Core)	4	100	50	50
1.2	Computer Fundamentals & Operating System	Major(Core)	2	50	0	50
1.3	Open Elective Course - I	OEC	4	100	50	50
1.4	Web Technology	VSC	2	50	50	0
1.5	Swayam/Chetana/MOOC	SEC	2	50	50	0
1.6		AEC	2	50	0	50
1.7		IKS	2	50	0	50
1.8		VEC	2	50	50	0
1.9	* Co-Curricular Course-I	CC	2	50	50	0
			22	550	300	250
<p>* 1.5. SWAYAM/CHETNA/MOOCs subjects should be from Skill Enhancement Course (SEC) category.</p> <p>* 1.6,1.7,1.8,1.9 Co-Curricular Course (Health & Wellness, Yoga education, sports & fitness, Cultural activities, NSS, NCC and Fine/applied/visual/performing arts) will be provide by The University</p>						
Semester II						
2.1	Programming Methodology using C++	Major(Core)	4	100	50	50
2.2	Digital Electronics	Major(Core)	2	50	0	50
2.3	Open-Source Operating System and Applications	Minor Stream	2	50	0	50
2.4	Open Elective Course – II	OEC	4	100	50	50
2.5	Introduction to Computer Hardware	VSC	2	50	0	50
2.6	Swayam/CHETNA/MOOC	SEC	2	50	50	0
2.7		AEC	2	50	50	0
2.8		VEC	2	50	0	50
2.9	* Co-Curricular Course-II	CC	2	50	50	0
			22	550	250	300
<p>* 2.7,2.8,2.9 Co-Curricular Course (Health & Wellness, Yoga education, sports & fitness, Cultural activities, NSS, NCC and Fine/applied/visual/performing arts) will be provide by the University</p>						

Exit with UG Certificate in Computer Application with 10 extra credits (44+10 credits)

SN	Courses	Type of Course	Credits	Marks	Int	Ext
	Semester III					
3.1	Java Programming	Major(Core)	4	100	50	50
3.2	Computer Organization and Architecture	Major(Core)	4	100	50	50
3.3	Database Management System	Minor Stream	4	100	50	50
3.4	Open Elective Course-III	OEC	2	50	0	50
3.5	Data Analytics using Spreadsheet	VSC	2	50	50	0
3.6		AEC	2	50	0	50
3.7	Field projects	FP	2	50	50	0
3.8	* Co-Curricular Course-III	CC	2	50	50	0
			22	550	300	250
*3.6 and 3.8 will be provide by the University *3.7 Field Project will be projects assigned to individual student on major subjects.						
	Semester IV					
4.1	Data Structure and Algorithm	Major(Core)	4	100	50	50
4.2	Introduction to Microprocessor	Major(Core)	4	100	50	50
4.3	Software Engineering	Minor Stream	4	100	50	50
4.4	Open Elective Course-IV	OEC	2	50	0	50
4.5	Swayam/Chetana/MOOC	SEC	2	50	0	50
4.6		AEC	2	50	0	50
4.7	Electronic Waste Management	CEP	2	50	50	0
4.8	Co-Curricular Course-IV	CC	2	50	50	0
			22	550	250	300
*4.6 and 4.8 will be provide by the University						

Exit with UG Diploma in Computer Application with 10 extra credits (44+10 credits)

OEC (Open Elective Course)	
Open Elective Course-I	Digital Marketing
Open Elective Course-II	Intellectual Property Rights
Open Elective Course-III	Computer Graphics
Open Elective Course-IV	Multimedia Computing

SN	Courses	Type of Course	Credits	Marks	Int	Ext
	Semester V					
5.1	Dot Net Technology	Major (Core)	4	100	50	50
5.2	Computer Networks	Major (Core)	4	100	50	50
5.3	Python Programming	Major (Core)	2	50	0	50
5.4	Elective - I	Major (Elective)	4	100	50	50
5.5	Introduction to Data Science	Minor Stream	4	100	50	50
5.6	Data Visualization	VSC	2	50	50	0
5.7	Field Project/Internship	FP/CEP	2	50	50	0
			22	550	300	250
	Semester VI					
6.1	Mobile Application Development using Android	Major (Core)	4	100	50	50
6.2	Cyber Security	Major (Core)	4	100	50	50
6.3	Cloud Computing	Major (Core)	2	50	0	50
6.4	Elective - II	Major (Elective)	4	100	50	50
6.5	Web 3.0 and Metaverse	Minor Stream	4	100	50	50
6.6	Internship/Apprenticeship	OJT	4	100	50	50
			22	550	250	300

Exit with Degree (3-year)

4-Year Degree with Honors

SN	Courses	Type of Course	Credits	Marks	Int	Ext
Semester VII						
7H.1	Data Warehousing and Data Mining	Major (Core)	4	100	50	50
7H.2	System Software	Major (Core)	4	100	50	50
7H.3	Software Testing and Quality Assurance	Major (Core)	4	100	50	50
7H.4	Software Testing and Quality Assurance Lab	Major (Core)	2	50	50	0
7H.5	Elective-III	Major (Elective)	4	100	50	50
7H.6	Research Methodology	Minor Stream(RM)	4	100	50	50
			22	550	300	250
Semester VIII						
8H.1	Big Data Analytics	Major (Core)	4	100	50	50
8H.2	Image Processing	Major (Core)	4	100	50	50
8H.3	Theory of Computation	Major (Core)	4	100	50	50
8H.4	Compiler Design	Major (Core)	2	50	0	50
8H.5	Elective-IV	Major (Elective)	4	100	50	50
8H.6	Project/Internship	OJT	4	100	50	50
			22	550	250	300

Elective	
Elective-I	Advance Databases
Elective-II	Artificial Intelligence
Elective-III	Machine Learning
Elective-IV	Internet of Things

4-Year Degree with Research

SN	Courses	Type of Course	Credits	Marks	Int	Ext
	Semester VII					
7R.1		Major (Core)	4	100	50	50
7R.2		Major (Core)	4	100	50	50
7R.3		Major (Core)	2	50	0	50
7R.4		Major (Elective)	4	100	50	50
7R.5		Minor Stream(RM)	4	100	50	50
7R.6		Research Project	4	100	100	0
			22	550	300	250
	Semester VIII					
8R.1		Major(Core)	4	100	50	50
8R.2		Major(Core)	4	100	50	50
8R.3		Major(Core)	2	50	0	50
8R.4		Major (Elective)	4	100	50	50
8R.5		Research Project	8	100	100	100
			22	550	250	300

Course Syllabus

Semester: I

1.1 Major (Core)

Course Title	PROBLEM SOLVING USING C
Course Credits	2
Course Outcomes	After Completion of this Course, students will be able to <ol style="list-style-type: none">1. Apply logic to create programs in C.2. Analyze and understand computer programming language concepts.3. Evaluate and interpret the use of pointers, including their declarations, initialization, and operations.4. Design and develop applications using basic programming constructs, facilitating the transition to other languages.
Module1 (Credit1)	
Learning Outcomes	After learning this module, learners will be able to <ol style="list-style-type: none">1. Learn steps in problem solving using C.2. Understand structure, keywords, operators, and functions of C programming.3. Evaluate the concept of I/O functions, header files, and preprocessor directives.4. Design and apply concepts of the C language.
Content Outline	<ul style="list-style-type: none">• Introduction to problem solving: Concept: Steps in problem solving - (Define Problem, Analyze Problem, Explore Solution), Problem solving techniques - (Trial & Error, Brain Storming, Divide & Conquer), Algorithms and Flowcharts (Definitions, Characteristics, Advantage & Disadvantages, Symbols, Examples), Pseudo-code (Definition, Conditional statements, Loops),etc.• Overview of programming languages: Definition of the program, Concept- Source code, Object code, Compilation, Interpretation, Execution, Input and Output, Debugging etc., Expressions, control structures; sub routines, Storage management; scoping rules; bindings for names, Storage types: Automatic, external, register and static variables• Introduction to 'C' Language: History of C Programming, Structures of „C“ , Programming, Simple example, Basic Input/ Output, Function as building blocks.Language Fundamentals:Character set,CTokens,Keywords, Identifiers,Variables, Constant, Data Types, Comments

	<ul style="list-style-type: none"> • Operators: Types of operators, Precedence and Associativity, Expression. Statement and types of statements, Built-in Operators and function. Console based I/O and related built in I/O Function: printf(), scanf(), getch(), getchar(), putchar(), etc; Concept of header files, Preprocess ordirectives: #include,#define, Conditional statements and Loops
Module2 (Credit1)	
Learning Outcomes	After learning the module, learners will be able to
	1. Gain proficiency in writing C programs to solve various problems.
	2. Learn the syntax and semantics of the C language, including its specific features such as pointers and memory management.
	3. Analyze the difference between structure and union.
	4. Design and handle operations of the files.
Content Outline	<ul style="list-style-type: none"> • Control structures <ul style="list-style-type: none"> ➤ Decision making structures: If, If-else, Nested If –else, Switch, Loop Control structures ➤ While, Do-while, For, Nested for, while, do-while loop, jumping statements: break, continue, go to, exit. • Functions: Definition, Basic types of function, Declaration and definition, Function call, Types of function, Parameter passing, Call by value, Call by reference, Scope of variables, Recursion, String: Declaration, string Functions, String Manipulations • Pointers: Introduction to pointers, Pointer notation, Pointer arithmetic, Null Pointer • Arrays: Definition, Declaration, Initialization, Bounds checking, One-Dimensional Array, Two-Dimensional Array, Passing array to a function, pointer to Array • Structure and Union: Introduction to Structure, Definition, Declaration of Structure Variables, Dot Operator, Nested Structure, Array of Structure, pointer to structure, Introduction to Union, Difference between Structure and Union • File Handling: Concept of File, Definition, File operations (create, open, read, move, write, close), File opening Mode, Closing a file, Input / output operations, Creating and reading a file, Command Line Argument

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1

1. Create a flowchart and algorithm for a simple problem (e.g., calculating the factorial of a number).
2. Write a pseudo-code for the above problem.
3. Convert the pseudo-code into a C program.
4. Demonstrate the use of basic input/output functions such as printf() and scanf() in the program.
5. Include the use of variables, constants, and data types.
6. Write a C program to demonstrate the use of different operators (arithmetic, relational, logical, bitwise, etc.).

7. Create examples to illustrate the precedence and associativity of operators and evaluate expressions.
8. Include conditional statements and loops in the program to show complex expressions and their evaluations.
9. Demonstrate the use of console-based I/O functions such as printf(), scanf(), getch(), getchar(), putchar(), etc.
10. Illustrate the use of header files and preprocessor directives (#include, #define).

Module 2

1. Write C programs using different control structures (if, if-else, switch, while, do-while, for loops). Include programs that utilize nested loops and jumping statements (break, continue, go to).
2. Create programs that use functions to perform various tasks. Include examples of parameter passing (call by value and call by reference).
3. Write a program that includes recursion and demonstrates string manipulation using string functions.
4. Develop a program that uses pointers for arithmetic operations and demonstrates the concept of null pointers.
5. Write a program to handle arrays (one-dimensional and two-dimensional) and pass them to functions. Include pointer to array operations.
6. Write programs to perform basic file operations (create, open, read, write, close). Include programs that demonstrate reading from and writing to files.
7. Implement a program that uses command-line arguments for file operations.
8. Compare and contrast the use of structures and unions in handling data through a practical example in a program.

References:-

1. Schildt, H. (2000). C: The Complete Reference (4th ed.). Tata McGraw-Hill Education Pvt. Ltd.
2. Ramkumar, & Agrawal. (1996). Programming in ANSI C. Tata McGraw-Hill.
3. Kanetkar, Y. P. (2008). Let Us C. Infinity Science Press.

1.1 Major (Core)

Course Title	PROBLEM SOLVING USING C (LAB)
Course Credits	2
Course Outcomes	After completion of this Course, the students will be able to
	1. Apply algorithms by writing C code.
	2. Analyze and trace the execution of C programs.
	3. Evaluate the use of pointers, arrays, and the pre-processor in programs.
	4. Design programs that utilize derived data types and implement simple file operations.
Module1 (Credit 1)	
Learning Outcomes	After learning this Module, learners will be able to
	1. Apply operators to write simple programs.
	2. Analyze and use control, iterative, and jumping statements.
	3. Evaluate the use of break and continue statements.

	4. Design programs with header files and preprocessor directives.
Content Outline	<ul style="list-style-type: none"> • Simple Program • Implementation of Operators: Built in Operators and function, Arithmetic, Logical, Relational, bitwise, Precedence And Associativity, composite statements. Unary, binary and ternary operators. • Concept of header files, Preprocessor directives: #include, #define. And macros implementations, Implementation of Storage types: Automatic external, register and static variables • Console based I/O and related built in I/O function: printf(), scanf(), getch(), getchar(), putchar(); • Control Statement: Decision Making Statements, if, Nested if, if-else, Nested if-else, if-else-if, switch, etc. The Conditional Expression; • Iterative Statements- The for loop, . The while loop, The do-while loop; Jumping Statements- The goto & label, The break & continue, The exit()function
Module 2 (Credit 1)	
Learning Outcomes	After learning this Module, learners will be able to
	1. Apply functions in programs.
	2. Analyze the declaration, initialization of pointers, and passing arrays to functions.
	3. Evaluate the definitions and declarations of structure variables in programs.
	4. Design programs effectively using functions, pointers, and structures.
Content Outline	<ul style="list-style-type: none"> • Implementation of Functions: Defining and accessing, passing arguments, Function prototypes, function calling mechanism, call byvalue, call byreference, recursive function; String Manipulations • Pointer Declaration and Initialization of Pointer variables, pointer Arithmetic, Pointers and Character Strings Implementation of 1-D and multidimension Array, One-Dimensional Array, Two-DimensionalArray, Passing array to a function, pointer to Array. • Programs Using Structure and Union: Defining and Declaring Structure Variables, .Dot Operator, Nested Structure, Array of Structure, pointer to structure, Examples of Union. • Programs using I/O Operations File Handling: File Operations (Create, open, read, move, write, close) Input/output operations on file Character by -(fgetc, fputc), Reading and writing files

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1

Write a C program that implements a simple calculator. The program should prompt the user to enter two numbers and an operator (+, -, *, /). Based on the operator entered, perform the corresponding arithmetic operation and display the result.

1. Use appropriate control flow statements (if, else if, else or switch) to determine the operation to be performed based on the operator entered by the user.
2. Implement error handling to handle division by zero.
3. Utilize console-based input/output functions like printf(), scanf() for user interaction.
4. Make use of header files and preprocessor directives to organize your code and define any necessary constants.

Module 2

Write a C program that demonstrates the use of functions, pointers, and structures to manage student records. Each student record should contain the following information: name, roll number, marks in three subjects.

1. Define a structure to represent a student record with appropriate data members.
2. Implement functions to perform the following tasks:
3. Input student details (name, roll number, marks).
4. Calculate the total marks and average marks of a student.
5. Display student details along with total and average marks.
6. Use pointers to pass structures to functions wherever necessary.
7. Ensure proper memory allocation and deallocation.
8. Implement file handling operations to read and write student records to a file using input/output functions like `fscanf()`, `fprintf()`, `fopen()`, `fclose()`, etc.

References:

1. Balagurusamy, E. (1990). C programming. Tata McGraw Hill.
2. Schildt, H. (2000). C: The Complete Reference (Fourth Edition). Tata McGraw-Hill Education Pvt. Ltd.
3. Ramkumar & Agrawal. (1996). Programming in ANSI C. Tata McGraw Hill.
4. Kanetkar, Y. P. (2008). Let Us C. Infinity Science Press.

Semester: I

1.2. Major (Core)

Course Title	COMPUTER FUNDAMENTALS AND OPERATING SYSTEM
Course Credits	2
Course Outcomes	<p>After Completion of this Course, students will be able to</p> <ol style="list-style-type: none"> 1. Apply knowledge to set up and connect various computer peripherals and interfaces. 2. Analyze and perform conversions between different number systems and execute binary arithmetic operations. 3. Evaluate and troubleshoot basic networking concepts and protocols. 4. Design and optimize operating system components to enhance their functions and overall system performance.
Module 1 (Credit 1)	
Learning Outcomes	<p>After learning the module, learners will be able to</p> <ol style="list-style-type: none"> 1. Apply concepts of number systems and computer arithmetic to solve computational problems. 2. Analyze the differences and functionalities of system and application software, and various types of computers. 3. Evaluate the characteristics and interpretations of data, including memory devices and computer languages. 4. Design efficient process scheduling and synchronization mechanisms within various types of operating systems.
Content Outline	<ul style="list-style-type: none"> • Number Systems: Binary, Octal, Decimal, Hexadecimal and Their inter conversion, Computer Arithmetic. • Computer Software: System and Application Software. • Type of Computers: Digital, Analog, Hybrid Computers • Definition: Data, Information; Characteristics and Interpretation, Data & its logical & physical Concepts, Definition of Computer, Features, Block Diagram of Computer System, Computer Generations • Primary Memory Devices: RAM, ROM, PROM, EPROM, CACHE Memory, Registers. • Computer Languages: Machine, Assembly, High Level • Operating System: Purpose of Operating Systems, OS Structure, Services of Operating System, Types of Operating System • Processes: Concept, process states, Scheduling, Operations on Processes, Cooperating Process, Process Synchronization.
Module 2 (Credit 1)	
Learning Outcomes	<p>After learning the module, learners will be able to</p> <ol style="list-style-type: none"> 1. Apply CPU scheduling algorithms (FCFS, SJF, RR, Priority) to optimize process scheduling. 2. Analyze various memory management techniques, including swapping, contiguous memory allocation, paging, and segmentation. 3. Evaluate different page replacement policies (LRU, OPT, SC, FIFO, NRU, MRU) to enhance memory management efficiency. 4. Design memory management strategies incorporating advanced scheduling and page replacement algorithms to improve overall system performance.

Content Outline	<ul style="list-style-type: none"> • CPU Scheduling: Concept, Scheduling Criteria, Scheduling Algorithms (FCFS, SJF, RR, Priority). • Memory Management: Concept, Swapping, Contiguous Memory Allocation, Paging, Segmentation. • Page Replacement policies: Least Recently used(LRU) Optimal (OPT), Second Chance (SC), First in First Out (FIFO), Not recently used (NRU), Most Recently Used(MRU).
------------------------	---

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

1. – **Module 1: Comprehensive Computer Systems Analysis and Implementation**
 - **Number Systems and Computer Arithmetic**
 - Convert the following numbers from one number system to another:
Binary to Decimal: 101011
Decimal to Hexadecimal: 175
Octal to Binary: 257
 - Perform the following binary arithmetic operations:
1011 + 1101
1001 - 0110
 - **Computer Software**
Compare and contrast system software and application software by listing at least 3 examples of each. Explain the primary functions of each type of software.
 - **Types of Computers**
Describe the characteristics and uses of digital, analog, and hybrid computers. Provide one real-world example for each type.
 - **Data and Computer Systems**
 - Define the terms data and information. Explain the difference between the two with examples.
 - Draw a block diagram of a computer system and label its main components.
 - Describe the features and characteristics of each generation of computers from the first to the current generation.
 - **Primary Memory Devices**
Explain the differences between RAM, ROM, PROM, EPROM, and CACHE memory. Include their primary uses and characteristics.
 - **Computer Languages**
 - Differentiate between machine language, assembly language, and high-level languages. Provide one example for each and explain their typical uses in programming.
 - **Operating Systems**
 - Describe the primary purpose and services provided by an operating system.
 - Explain the structure of an operating system with a diagram.
 - List and briefly describe different types of operating systems (e.g., batch, time-sharing, distributed).
 - **Processes and Scheduling**
 - Explain the concept of a process and its states. Illustrate the process state diagram.
 - Describe different CPU scheduling algorithms (FCFS, SJF, RR, Priority). Provide a scenario where each algorithm would be most appropriate.
 - Explain process synchronization and provide an example where it is crucial in an operating system.

2. – **Module 2: To apply, analyze, evaluate, and design CPU scheduling algorithms, memory management techniques, and page replacement policies through a comprehensive simulation and analysis activity.**

CPU Scheduling:

- **Define and explain the concepts of CPU scheduling and scheduling criteria.**
Implement and compare the following scheduling algorithms: FCFS, SJF, RR, and

Priority.

- Task: Simulate a scenario with a set of processes, each having different burst times, arrival times, and priorities. Evaluate the performance of each scheduling algorithm based on average waiting time, turnaround time, and CPU utilization.
- Deliverable: Provide a detailed report with your simulation code, a comparison of the results, and an analysis of which algorithm performs best under different conditions.

- **Memory Management:**

Explain the concepts of memory management, including swapping, contiguous memory allocation, paging, and segmentation.

- Task: Simulate a memory management system that includes swapping and both contiguous and non-contiguous allocation methods (paging and segmentation).
- Deliverable: Create a memory map for each allocation method. Provide your simulation code, memory maps, and an analysis of the advantages and disadvantages of each method.

- **Page Replacement Policies:**

Define and explain the following page replacement policies: LRU, OPT, SC, FIFO, NRU, and MRU.

- Task: Simulate a scenario where a sequence of page requests is given. Implement each of the page replacement policies and evaluate their performance based on the number of page faults.
- Deliverable: Provide your simulation code, a comparison of the number of page faults for each policy, and an analysis of which policy is most effective under different conditions.

References:

1. Stallings, W. (2021). Computer organization and architecture: Designing for performance (11th ed.). Pearson Education.
2. Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). Operating system concepts (10th ed.). Wiley.
3. Tanenbaum, A. S., & Austin, T. (2012). Structured computer organization (6th ed.). Pearson Education.
4. Patterson, D. A., & Hennessy, J. L. (2017). **Computer organization and design: The hardware/software interface** (5th ed.). Morgan Kaufmann.
5. Stallings, W. (2018). **Operating systems: Internals and design principles** (9th ed.). Pearson Education.

Semester: I

1.3 OEC (Open elective Course-I)

Course Title	DIGITAL MARKETING
Course Credits	4
Course Outcomes	<p>After completion of this Course, the students will be able to</p> <ol style="list-style-type: none"> 1. Apply principles and foundations of digital marketing. 2. Analyze data to optimize marketing strategies and campaigns. 3. Evaluate digital marketing performance using analytics tools. 4. Design effective marketing using digital channels, customer insights, and segmentation.
Module1 (Credit1)	
Learning Outcomes	<p>After learning this module, learners will be able to</p> <ol style="list-style-type: none"> 1. Apply principles and techniques of digital marketing using various channels and tools. 2. Analyze and understand SEO techniques. 3. Evaluate the effectiveness of digital marketing strategies. 4. Design comprehensive digital marketing campaigns.
Content Outline	<ul style="list-style-type: none"> • Basics of Websites & Digital Marketing: - <ul style="list-style-type: none"> ➤ Fundamental of Digital Marketing: Concept, Scope, Areas to Explore ➤ Building Websites on different content management system like Weebly, WordPress & Blogger ➤ Designing: Canva Tool for Image Editing and Photoshop Graphics • SEO -On Page Optimization: - <ul style="list-style-type: none"> ➤ Broken links, Backlinks, W3 Errors, Keyword research & optimization Heading Tag Optimization: Reporting, Suggestion and Implementing Backlinks, Titles & Meta Descriptions, Website Content Optimization Meta& Title Tags Adjustment, XLM Site Map Setup, Robot.txt Validation Google Analytics and Webmaster Tool Setup ➤ SERP - rankings on google, Plugins Installation and Monitoring Heading Tag Optimization: Reporting, Suggestion and Implementing Duplicate Content Reporting ➤ Duplicate Content Rewording/rewriting using seo target keywords ➤ Plugins & Internal Linking, Permalinks Optimization: Reporting, Suggestion and Implementing
Module2 (Credit1)	
Learning Outcomes	<p>After learning this module, learners will be able to</p> <ol style="list-style-type: none"> 1. Apply strategies for effective campaigns. 2. Analyze website metrics using Google Analytics. 3. Evaluate important metrics for SEM campaigns. 4. Design and optimize campaigns based on analytics data.
Content Outline	<ul style="list-style-type: none"> • Social Media Optimization: - Custom Graphics and Setting Profile with about/ hours and other information of business: For your Social Media Account
Module3 (Credit1)	
Learning Outcomes	<p>After learning this module, learners will be able to</p> <ol style="list-style-type: none"> 1. Apply business analytics techniques for data-driven marketing decisions.

	2. Analyze data insights to optimize marketing strategies and campaigns.
	3. Evaluate the effectiveness of data-driven marketing decisions.
	4. Design marketing strategies based on interpreted data insights.
Content Outline	<ul style="list-style-type: none"> • Social Media Content Creation and Posting: Social Media Platform: one posts for each platform and two Image Post and one interesting post found related to the business you are in on web and shared on your page each day. Along with: Commenting, Follows, Likes, Shares • Paid Advertising: Email Marketing, Social media Marketing and AdWords : Ad Plan, Ad Setup with Banner Images, Monitoring & Reporting (1 Ad: Website Conversion, Apps Installation, Promote Page Etc.) on website AdWords: Search Ads Type: Keyword Research, 1Campaign, 2Adgroups, 5Ads.DisplayAds: 2Banner Graphics keyword Research, 1AdGroup, 2 Ads. • Email Marketing: Content Writing as per Offers, services & Discounts, Custom Template Building, HTML Conversion, Use of Emailing Software: Account Creation, Creating Customer List, Sending Emails
Module4 (Credit1)	
Learning Outcomes	After learning this module, learners will be able to
	1. Apply appropriate metrics and analytics tools to measure digital marketing campaign performance.
	2. Analyze data gathered from campaigns to evaluate their effectiveness.
	3. Evaluate the performance of digital marketing campaigns based on measured metrics.
	4. Design strategies for optimizing digital marketing campaign performance based on evaluation results.
Content Outcomes	<ul style="list-style-type: none"> • Video sharing platform Optimization: Posting, pinning, repining Analytics: Search engine Analytics, Installing Analytics, How to Study search engine Analytics, Interpreting Bars & Figures, How Analytics can Help SEO, Advanced Reporting, Webmaster Central, Bing/Yahoo, Open Site Explorer, Website Analysis using various SEO Tools available. • Reporting and Monitoring: - SEO REPORT/plan, Initial website ranking and evaluation report Analytics report and web master report Ad Words Report, Fb Insight, Marketing Media Audit Reports ERP Report, Competitor Analysis Report(Media Audit Report)

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1:

Design and execute an SEO audit for a website.

1. Perform on-page optimization tasks such as checking for broken links, W3 errors, keyword research, optimizing heading tags, titles, and meta descriptions.
2. Implement necessary changes based on the audit findings, including fixing broken links, optimizing content, and adjusting meta tags.
3. Set up essential tools like Google Analytics and Google Webmaster Tools for monitoring website performance and tracking SEO metrics.

4. Create a comprehensive report summarizing the audit findings, implemented changes, and recommendations for further optimization.

Module 3:

Develop a social media marketing campaign for a business.

1. Create custom graphics and optimize social media profiles for the selected business on major platforms like Facebook, Twitter, Instagram, etc.
2. Plan and execute a paid advertising campaign using platforms like Google AdWords and social media ads.
3. Design and send email marketing campaigns, including content creation, template building, and email list management.
4. Monitor and analyze the performance of the campaigns using relevant metrics and analytics tools.
5. Prepare detailed reports showcasing campaign performance, including insights, key metrics, and recommendations for improvement.

References

1. Chaffey, D., & Ellis-Chadwick, F. (2019). *Digital Marketing: Strategy, Implementation and Practice* (7th ed.). Pearson Education Limited.
2. Ryan, D. (2017). *Understanding Digital Marketing: Marketing Strategies for Engaging the Digital Generation* (4th ed.). Kogan Page.
3. Kotler, P., Kartajaya, H., & Setiawan, I. (2017). *Marketing 4.0: Moving from Traditional to Digital*. Wiley.

Semester: I

1.4. VSC

Course Title	WEB TECHNOLOGY
Course Credits	2
Course Outcomes	After completion of this Course, the students will be able to
	1. Apply fundamental concepts of web technology, including the World Wide Web, HTTP protocol, web browsers, and web servers.
	2. Analyze HTML and CSS proficiency to create visually appealing web pages.
	3. Evaluate the use of JavaScript to enhance interactivity and add dynamic functionality to web pages.
	4. Design dynamic web applications by implementing server-side scripting languages and integrating them with databases for data storage and retrieval.
Module1 (1 Credit)	
Learning Outcomes	After learning this module, learners will be able to
	1. Apply deep understanding of web technology concepts, protocols, and standards.
	2. Analyze HTML and CSS proficiency to create visually appealing web pages.
	3. Evaluate interactive and dynamic web page development using JavaScript and DOM manipulation.
	4. Design and implement server-side scripts for handling user requests and generating dynamic web content.
Content Outline	<ul style="list-style-type: none">• Introduction to Web Technology: World Wide Web, HTTP protocol, web browsers, and web servers.• HTML Basics: Document structure, tags, elements, and attributes.• Cascading Style Sheets (CSS): Styling webpages, selectors, properties, and positioning.• Responsive Web Design: Creating web pages that adapt to different screen sizes.• Java Script Fundamentals: Variables, datatypes, operators, control structures, and functions.• Document Object Model (DOM): Manipulating web page elements using Java Script.

Module2 (Credit 1)	
Learning Outcomes	After learning this module, learners will be able to
	1. Apply integration of databases with web applications for efficient data handling.
	2. Analyze and implement secure coding practices to safeguard web applications.
	3. Evaluate the effectiveness of web development frameworks and tools in streamlining processes.
	4. Design database integration methods for optimizing data storage and retrieval in web applications.
Content Outline	<ul style="list-style-type: none"> • Client-Side Scripting: Event handling, form validation, and dynamic content generation using JavaScript. • Introduction to Web Development Frameworks (e.g., React, Angular, Vue.js) • Server-Side Scripting: Introduction to server-side scripting languages (e.g., Python, Ruby). • Dynamic Web Pages: Generating dynamic content based on user input and server-side processing. • Database Connectivity: Introduction to databases, SQL queries, and database integration with web applications.

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1

Create a static website using HTML, CSS, and JavaScript.

1. Design a multi-page website with a consistent layout and navigation menu.
2. Use HTML to structure the content of each webpage, including appropriate tags, elements, and attributes.
3. Apply CSS to style the website, including selectors, properties, and responsive design techniques for different screen sizes.
4. Implement JavaScript to add interactivity to the website, such as dynamic content manipulation and form validation.
5. Ensure the website adheres to web standards and best practices for accessibility and usability.

Module 2

Develop a web application with database integration and secure coding practices.

1. Choose a web development framework (e.g., React, Angular, Vue.js) and use it to build a dynamic web application.
2. Implement client-side scripting using JavaScript for event handling, form validation, and dynamic content generation.
3. Use server-side scripting languages (e.g., Python, Ruby) to handle user requests and interact with a backend database.
4. Design and implement database connectivity, including creating SQL queries for data manipulation and integration with the web application.
5. Apply secure coding practices to safeguard the web application against common vulnerabilities such as XSS (Cross-Site Scripting) and SQL injection.
6. Test the web application thoroughly to ensure functionality, security, and performance.

References: -

1. Lecky-Thompson, G. W. (2009). Web Programming. Cengage Learning.
2. Powell, T. (2000). Web Design: The Complete Reference. Tata McGraw Hill.
3. Powell, T. (2008). HTML and XHTML: The Complete Reference. Tata McGraw Hill.
4. Powell, T., & Schneider, F. (2004). JavaScript 2.0: The Complete Reference (2nd ed.). Tata McGraw Hill.
5. Holzner, S. (2017). PHP: The Complete Reference. Tata McGraw Hill.

Semester: I

1.5 SEC (SWAYAM/CHETNA/MOOCs)

Semester: I

1.6. AEC

Semester: 1

1.7 IKS

Semester: I

1.8 VEC

Semester: I

1.9 CC (Co-curricular Course-I)

Semester: II

2.1 Major (Core)

Course Title	PROGRAMMING METHODOLOGY USING C++
Course Credits	2
Course Outcomes	After completion of this Course, the students will be able to <ol style="list-style-type: none">1. Apply object-oriented programming concepts in C++.2. Analyze problems and develop C++ programs to solve them.3. Evaluate the use of file input/output in C++.4. Design solutions using object-oriented programming principles in C++.
Module1 (Credit1)	
Learning Outcomes	After learning the module, learners will be able to <ol style="list-style-type: none">1. Apply fundamental programming concepts including variables, data types, control structures, functions, arrays, and objects.2. Analyze and implement object-oriented programming concepts like objects, classes, and defining functions and variables.3. Evaluate the use of object-oriented programming in solving programming problems.4. Design solutions using a combination of fundamental programming and object-oriented concepts.
Content Outline	<ul style="list-style-type: none">• Evolution of OOP: Advantages and disadvantages of OOP over its predecessor paradigms, Characteristics of Object-oriented Programming: Abstraction, Encapsulation, Data hiding, Inheritance, Polymorphism, Code Extensibility and Reusability, User defined Data Types.• C++Program Structure, Simple Input/ Output Program, Program Comments, Identifiers, Literals, String, Character, Integer, Floating Point, Constants, Keywords, Data Types, Operators in C++, Control Structures in C++.• Object and Classes: Core object concepts, Encapsulation, Abstraction, Polymorphism, Classes, Messages Association, Interfaces, Implementation of class in C++, C++ Objects as physical object, C++ object as data types constructor Object as function arguments.• Functions and Variables: Functions: Declaration and Definition, Variables: Definition Declaration, and Scope, Dynamic Creation and Derived Data, Arrays and Strings in C++.

Module2 (Credit1)	
Learning Outcomes	After learning the module, learners will be able to
	1. Apply concepts of constructors, inheritance (including its types), and polymorphism in C++.
	2. Analyze file input/output handling in C++ and class templates.
	3. Evaluate the effectiveness of constructors, inheritance, and polymorphism in solving programming tasks.
	4. Design solutions involving file input/output operations and the utilization of class templates in C++.
Content Outline	<ul style="list-style-type: none"> • Inheritance: Concept of Inheritance, Derived class and base class, Types of Inheritance, Functions and Friend Functions. • Constructors: Multiple Constructors and Initialization, Using Destructors to Destroy Instances. • Polymorphism: Syntax for Operator overloading, overloading of unary and binary operators. • File input and output: Reading a File, Managing I/O Streams, opening a File – Different Methods, Checking for Failure with File Commands • Class templates: Implementing a class template, implementing class template member functions, Using a class template, Function templates

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1:

Develop a console-based application in C++ that demonstrates the implementation of fundamental programming concepts and object-oriented principles.

1. Create a C++ program that includes classes representing real-world entities (e.g., a student, a car, a bank account).
2. Implement basic functionality within these classes, such as setting and retrieving object attributes, defining member functions for data manipulation, and demonstrating encapsulation and abstraction.
3. Utilize inheritance to create derived classes that inherit properties and behaviors from base classes, showcasing the concept of code reusability.
4. Incorporate polymorphism by defining virtual functions and overriding them in derived classes to demonstrate runtime polymorphism.
5. Implement file input/output operations to store and retrieve data related to objects, showcasing the handling of persistent data using C++.
6. Document your code thoroughly and provide comments to explain the purpose and functionality of each component.

Module 2 :

Design and implement a template-based class hierarchy in C++ for managing a generic data structure.

1. Define a base template class that represents a generic data structure (e.g., a linked list, a stack, a queue).
2. Implement derived template classes that inherit from the base class and specialize it to handle specific data types or functionalities (e.g., a linked list of integers, a stack of strings).
3. Utilize constructor overloading to provide flexibility in initializing instances of the template classes.
4. Implement operator overloading to enable intuitive manipulation of objects within the class hierarchy (e.g., addition, subtraction for mathematical operations).
5. Use file input/output operations to demonstrate the serialization and deserialization of objects of the template classes.
6. Test your implementation with various data types and scenarios to ensure correctness and functionality.
7. Provide comprehensive documentation explaining the design decisions, implementation details, and usage instructions for the template-based class hierarchy.

References:

1. Balaguruswamy, E. (2008). Object Oriented Programming with C++. Tata McGraw-Hill Education.
2. Venugopal, K. R. (1997). Mastering C++. Tata McGraw-Hill Education.
3. Stroustrup, B. (1997). C++ Programming Language (3rd ed.). Addison Wesley.
4. Chandra, B. (1998). A Treatise On Object Oriented Programming using C++. Narosa Publications.
5. Schildt, H. (2001). The Complete Reference CN. Tata McGraw-Hill.

2.1 Major (Core)

Course Title	PROGRAMMING METHODOLOGY USING C++ (LAB)
Course Credits	2
Course Outcomes	After completion of this Course, the students will be able to <ol style="list-style-type: none">1. Apply object-oriented programming concepts by creating programs with classes and objects in C++.2. Analyze the implementation of object-oriented programming concepts in C++.3. Evaluate the effectiveness of stream I/O and file I/O in developing applications.4. Design object-oriented programs using templates and exceptional handling techniques in C++.
Module1 (Credit1)	
Learning Outcomes	After learning this Module, Learners will be able to <ol style="list-style-type: none">1. Apply the learned concepts by writing simple programs using classes and objects in C++.2. Analyze the implementation of object-oriented programming concepts in C++ to understand their functionality and usage.3. Evaluate the effectiveness of object-oriented programming in solving programming tasks.4. Design solutions using object-oriented programming concepts to improve code structure and modularity in C++.
Content Outline	<ul style="list-style-type: none">• Simple Programs on fundamental Data Types and I/O operators: Derived data types, Symbolic constants, variables and Reference variables• Operators and decision control structures: Programs to implement if statements, Switch statements, Loop statements and Functions in C++
Module2 (Credit1)	
Learning Outcomes	After learn in this Module, Learners will be able to <ol style="list-style-type: none">1. Apply the concepts of inheritance, constructors, and operator overloading by writing simple programs in C++.2. Analyze the process of reading and creating text files using C++ programs.3. Evaluate the effectiveness of inheritance, constructors, and operator overloading in solving programming tasks.4. Design programs that utilize inheritance, constructors, and operator overloading for improved code organization and functionality in C++.

Content Outline	<ul style="list-style-type: none"> • Programs to implement Object and Classes, Friend Function, Inheritance, Constructor and Destructor, Polymorphism, Overloading (Unary, Binary, Using friend functions etc.) • Files and streams • Class templates: Implementations of Class template, Class template with multiple parameters, Function template.
------------------------	--

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1:

Develop a set of C++ programs that demonstrate the fundamental concepts of object-oriented programming (OOP).

1. Write a C++ program that utilizes classes and objects to represent real-world entities (e.g., a car, a student, a bank account).
2. Implement simple programs to demonstrate the usage of derived data types, symbolic constants, variables, and reference variables.
3. Create programs that utilize decision control structures such as if statements, switch statements, and loop statements to demonstrate control flow in C++.
4. Design and implement functions within classes to showcase encapsulation and abstraction principles.
5. Provide comments and documentation within your code to explain the purpose and functionality of each component.

Module 2 :

Design a series of C++ programs that utilize inheritance, constructors, operator overloading, and file handling.

1. Implement a program that demonstrates the concept of inheritance by creating a base class and derived classes that inherit from it, showcasing the reuse of code and the specialization of functionality.
2. Write programs that utilize constructors and destructors to initialize and clean up resources within objects, demonstrating the importance of proper resource management.
3. Implement operator overloading in C++ programs to demonstrate the ability to define custom behavior for operators such as +, -, *, etc.
4. Develop programs that read from and write to text files using file handling techniques in C++, showcasing the ability to manipulate external data.
5. Utilize class templates to create generic data structures or algorithms that can operate on different data types, demonstrating the power of template-based programming.
6. Test your programs with various scenarios and inputs to ensure correctness and functionality.
7. Provide comprehensive documentation explaining the design decisions, implementation details, and usage instructions for each program.

References/Textbooks:-

1. Balguruswamy, E. (2008). Object Oriented Programming with C++. Tata McGraw-Hill Education.
2. Venugopal, K. R. (1997). Mastering C++. Tata McGraw-Hill Education.
3. Stroustrup, B. (1997). C++ Programming Language (3rd ed.). Addison Wesley.
4. Chandra, B. (1998). A Treatise On Object Oriented Programming using C++. Narosa Publications.
5. Schildt, H. (2001). The Complete Reference CN. Tata McGraw-Hill.

Semester: II

2.2 Major (Core)

Course Title	DIGITAL ELECTRONICS
Course Credits	2
Course Outcomes	After completion of this Course, the students will be able to
	1. Apply knowledge of digital logic levels to analyze and design digital electronics circuits effectively.
	2. Analyse digital signals, positive and negative logic, Boolean algebra, and logic gates.
	3. Evaluate logical variables, truth tables, number systems, and codes, including their conversion methods.
	4. Design and implement applications of Boolean algebra and logic gates in practical digital electronics scenarios.
Module 1 (Credit 1)	
Learning Outcomes	After learning this Module, learners will be able to
	1. Apply knowledge of number systems and codes in digital systems, integrating with logic gates, combinational, and sequential circuits.
	2. Analyse specifications to design and implement digital systems, creating truth tables, designing logic circuits, simulating them, and validating functionality effectively.
Content Outline	<ul style="list-style-type: none"> • Number Systems and Codes: Review of Binary, Octal and Hexadecimal Number Systems – Conversion methods- complements- signed and unsigned Binary numbers. Binary codes: Weighted and non-Weighted codes – ASCII – Error detecting and Error correcting codes- hamming codes. • Computer Arithmetic: Integer Representation, Integer Arithmetic, Floating Point Representation, Floating Point Arithmetic, Sources of Errors, Propagated Errors. • Digital Logic Circuits: Introduction to digital signals, Logic Gates, Universal gates, Implementation of Universal gates using basic gates. Conversion of Universal gates into Basic Gates, Exclusive gates Truth table, De-Morgan's Theorem: Statement and Proof.
Module 2 (Credit 1)	
Learning Outcomes	After learning this Module, learners will be able to
	1. Apply Boolean algebra to manipulate and simplify logic functions, implementing logic circuits.
	2. Analyse the behavior and truth tables of different types of logic gates, such as AND, OR, and NOT.
	3. Evaluate various logic circuits using simplified Boolean expressions.
	4. Design complex digital systems by integrating multiple logic gates and Boolean functions.
Content Outline	<ul style="list-style-type: none"> • Boolean Algebra: Boolean Laws, Simplification of Boolean expression using Laws, Min terms (SOP) Max terms (POS), Standard/Canonical SOP and POS forms, K-map (2.3 and 4 variables) Don't care conditions. • Truth tables: Simplification of Boolean expression using Truth Tables

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1:

1. Number Systems and Codes:
 - Convert given numbers between Binary, Octal, and Hexadecimal systems.
 - Demonstrate the use of complements and signed/unsigned binary numbers in practical applications.
 - Implement and analyze various binary codes (weighted, non-weighted, ASCII) and error-detecting/correcting codes using Hamming codes.
2. Computer Arithmetic:
 - Perform integer and floating-point arithmetic operations, identifying and analyzing sources of errors and propagated errors in computations.
3. Digital Logic Circuits:
 - Design and simulate basic digital circuits using logic gates (AND, OR, NOT, NAND, NOR).
 - Implement universal gates using basic gates and vice versa.
 - Validate the design by constructing truth tables for the designed circuits.
 - Apply De Morgan's Theorem to simplify given logical expressions.

Module 2:

1. Boolean Algebra:
 - Simplify given Boolean expressions using Boolean laws and postulates.
 - Convert Boolean expressions to their standard/canonical SOP (Sum of Products) and POS (Product of Sums) forms.
 - Use Karnaugh Maps (K-maps) for 2, 3, and 4-variable expressions to simplify Boolean functions, including handling don't care conditions.
2. Truth Tables:
 - Construct truth tables for given Boolean expressions and simplify the expressions using the truth tables.
3. Complex Digital Systems Design:
 - Design a complex digital system integrating multiple types of logic gates (AND, OR, NOT, NAND, NOR, XOR, XNOR).
 - Implement the design in a simulation tool, create the corresponding truth table, and validate the functionality.

References:-

1. Jain, R. P. (2003). Modern digital electronics. Tata McGraw-Hill Education.
2. Palan, N. G. (1998). Logic circuit. Technova Publication.

Course Syllabus

Semester: II

2.3 Minor Stream

Course Title	OPEN SOURCE OPERATING SYSTEM AND APPLICATIONS
Course Credits	2
Course Outcomes	<p>After completion of this Course, the students will be able to</p> <ul style="list-style-type: none"> • Apply knowledge of new developments in Information Technology and Cyber laws in real-world scenarios. • Analyze new technologies and the basics of Cyber laws, including case studies, to understand their implications. • Evaluate the effectiveness of new technology and Cyber laws in addressing contemporary challenges. • Design strategies for integrating new developments in Information Technology and Cyber laws into existing frameworks, considering case studies for practical application.
Module1 (Credit1)	
Learning Outcomes	<p>After learning the module, learners will be able to</p> <ul style="list-style-type: none"> • Apply skills in installing and configuring Red Hat Linux, including disk space management. • Analyze the setup and management of the Apache web server to establish a solid foundation in Linux system administration and web server management. • Evaluate the configuration of a dual boot system to facilitate seamless switching between Red Hat Linux and other operating systems. • Design an effective Apache installation process and configuration to serve web content efficiently.
Content Outline	<ul style="list-style-type: none"> • INSTALLING RED HAT LINUX: Configuring a Dual Boot System, Allocating Disk Space for Linux, Add a new Hard Drive , Use an Existing Partition to Create Space for Loading Linux • Using fdisk to Partition a Hard Disk Viewing, The Current Partitions, Deleting Partitions, Creating New Partitions • The Apache Installation Process, Apache Configuration, Manipulating the Apache HTTP Service
Module2 (Credit1)	
Learning Outcomes	<p>After learning the module, learners will be able to</p> <ul style="list-style-type: none"> • Apply the installation process of PHP on a Red Hat Linux system, demonstrating understanding of the steps involved. • Analyze and utilize the Linux command line for installing MySQL RPMs based on preferences and system requirements. • Evaluate different methods for installing MySQL, including utilizing the Add/Remove Applications tool. • Design efficient procedures for installing PHP and MySQL on Red Hat Linux systems, considering system preferences and requirements.
Content Outline	<ul style="list-style-type: none"> • Installing PHP : Quick Install Of PHP, Starting the Install Process to Begin PHP Configuration, to complete Installation of PHP, Binding the PHP Installation with Apache, Registering the Changes made in the httpd conf With Apache • INSTALLING MySQL: Using the Add/Remove Applications Tool, Using the Linux Command Line, Installing the My SQL RPMS,

	what to do if the Error Cannot Be Handled Easily, The Directory Tree Created during Installation, MySQL DATABASE ENGINE INSTALL, MySQL Database administration
--	--

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1 :

Create a step-by-step guide for installing and configuring Red Hat Linux and the Apache web server.

1. Explain the process of installing Red Hat Linux, including disk space management and configuring a dual boot system.
2. Provide instructions for using fdisk to partition a hard disk, viewing current partitions, deleting partitions, and creating new partitions.
3. Describe the Apache installation process, including configuration and manipulation of the Apache HTTP service.
4. Include screenshots and examples to illustrate each step of the installation and configuration process.
5. Discuss best practices for allocating disk space, managing partitions, and optimizing Apache configuration for efficient web content serving.

Module 2:

Develop a tutorial for installing PHP and MySQL on a Red Hat Linux system.

1. Explain the installation process of PHP, including quick installation steps, configuration, and binding PHP with Apache.
2. Provide guidance for installing MySQL using both the Add/Remove Applications tool and the Linux command line.
3. Discuss different methods for installing MySQL RPMs and troubleshooting common errors that may arise during installation.
4. Describe the directory tree created during MySQL installation and provide instructions for MySQL database administration.
5. Include practical examples and demonstrations to help learners understand the installation and configuration steps effectively.

References:

1. Holzner, S. (2017). PHP: The Complete Reference. Tata McGraw Hill.
2. Clydebank Technology. (2015). SQL Quickstart Guide: The Simplified Beginner's Guide to SQL.
3. Rosen, K. H., Host, D. A., Farber, J., & Rosinski, R. R. (1999). UNIX: The Complete Reference. Osborne.

Semester: II

2.4 OEC (Open Elective Course-II)

Course Title	INTELLECTUAL PROPERTY RIGHTS
Course Credits	4
Course Outcomes	After going through the course, learners will be able to
	<ul style="list-style-type: none"> • Apply knowledge of Intellectual Property Rights to protect creative work effectively.
	<ul style="list-style-type: none"> • Analyze the use of Intellectual Property in various contexts.
	<ul style="list-style-type: none"> • Evaluate the effectiveness of using Intellectual Property to safeguard creative work.
Module1 (Credit1)	
Learning Outcomes	After learning the module, learners will be able to
	<ul style="list-style-type: none"> • Apply knowledge and understanding of the justifications and rationales for protecting intellectual property in practical scenarios.
	<ul style="list-style-type: none"> • Analyze the reasons behind the need to protect intellectual property rights.
	<ul style="list-style-type: none"> • Evaluate the effectiveness of intellectual property protection in promoting innovation and creativity.
Content Outline	<ul style="list-style-type: none"> • Basic Principles and Acquisition of Intellectual Property Rights: Philosophical Aspects of Intellectual Property Laws, Basic Principles of Patent Law, Patent Application procedure, drafting of a Patent Specification, Understanding Copyright Law, Basic Principles of Trade Mark, Basic Principles of Design Rights, International Background of Intellectual Property.
Module2 (Credit1)	
Learning Outcomes	After learning the module, learners will be able to
	<ul style="list-style-type: none"> • Apply knowledge and understanding of different countries' Intellectual Property Rights (IPR) acts in legal contexts.
	<ul style="list-style-type: none"> • Analyze the similarities and differences between various countries' IPR acts.
	<ul style="list-style-type: none"> • Evaluate the effectiveness of different countries' IPR acts in protecting intellectual property.
	<ul style="list-style-type: none"> • Design strategies for navigating and complying with different countries' IPR acts based on understanding and knowledge.

Content Outline	<ul style="list-style-type: none"> • Information Technology Related Intellectual Property Rights: Computer Software and Intellectual Property-Objective, Copyright Protection, Reproducing, Defences, Patent Protection. • Database and Data Protection-Objective, Need for Protection, UK Data Protection Act, 1998, US Safe Harbor Principle, Enforcement. • Protection of Semi-conductor Chips-Objectives, Justification of protection, Criteria, Subject matter of Protection, WIPO Treaty, TRIPs, SCPA. • Domain Name Protection- Objectives, domain name and Intellectual Property, Registration of domain names, disputes under intellectual Property Rights, Jurisdictional Issues, and International Perspective.
Module3 (Credit 1)	
Learning Outcomes	<p>After learning the module, learners will be able to</p> <ul style="list-style-type: none"> • Apply understanding of different patents and copyrights information to protect intellectual property. • Analyze the process of patenting and development to secure legal rights. • Evaluate the procedure of trademark development for branding and protection. • Design strategies for navigating the patenting, copyrighting, and trademark development processes effectively.
Content Outline	<ul style="list-style-type: none"> • Patents (Ownership and Enforcement): Objectives, Rights, Assignments, Defenses in case of Infringement. • Copyright (Ownership and Enforcement): Copyright: Objectives, Rights, Transfer of Copyright, work of employment Infringement, Defenses for infringement. • Trademark (Ownership and Enforcement): Trademarks: Objectives, Rights, Protection of goodwill, Infringement, Passing off, Defenses. Designs: Objectives, Rights, Assignments, Infringements, Defenses of Design Infringement.
Module4 (Credit1)	
Learning Outcomes	<p>After learning the module, learners will be able to</p> <ul style="list-style-type: none"> • Apply knowledge of new developments in Information Technology and Cyber laws in real-world scenarios. • Analyze new technologies and the basics of Cyber laws, including case studies, to understand their implications. • Evaluate the effectiveness of new technology and Cyber laws in addressing contemporary challenges. • Design strategies for integrating new developments in Information Technology and Cyber laws into existing frameworks, considering case studies for practical application.
Content Outline	<ul style="list-style-type: none"> • Enforcement of Intellectual Property Rights: Civil Remedies, Criminal Remedies, Border Security measures. Practical Aspects of Licensing: Benefits, Determinative factors, important clauses, licensing clauses. • Cyber Law: Basic Concepts of Technology and Law: Understanding the Technology of Internet, Scope of Cyber Laws, Cyber Jurisprudence Law of Digital Contracts: The Essence of Digital Contracts, The System of Digital Signatures, The Role and Function of Certifying Authorities, The Science of Cryptography. • Case studies: Case studies related to different cyber crimes and punishment can be given.

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1 :

Create a comprehensive report or presentation that explores the principles and acquisition of intellectual property rights.

1. Discuss the philosophical aspects of intellectual property laws and the basic principles of patent law, copyright law, trade mark law, and design rights.
2. Explain the international background of intellectual property and the significance of protecting intellectual property rights in promoting innovation and creativity.
3. Analyze the process of patent application, including drafting a patent specification, and discuss the basic principles of copyright and trademark law.
4. Design strategies to advocate for and support the protection of intellectual property rights, considering the justifications and rationales behind such protection.
5. Provide examples and case studies to illustrate key concepts and practical applications of intellectual property rights.

Module 2:

Develop a series of case studies and a presentation focusing on information technology-related intellectual property rights and cyber law.

1. Choose several case studies related to computer software and intellectual property, database and data protection, protection of semi-conductor chips, and domain name protection.
2. Analyze each case study, discussing the objectives, legal issues, enforcement measures, and international perspectives.
3. Explore the concepts of patents, copyrights, trademarks, and designs in the context of ownership, enforcement, rights, and infringement.
4. Discuss the basics of cyber law, including the scope of cyber laws, cyber jurisprudence, digital contracts, digital signatures, cryptography, and relevant case studies.
5. Design strategies for navigating the complexities of intellectual property rights and cyber law in the context of new developments in information technology, considering practical applications and enforcement measures.

Reference:

1. Sood, V. (2017). Cyber Law. McGraw Hill Education
2. Leland, C. R. (1995). Licensing Art & Design. Allworth Press.
3. Leland, C. R. (1995). A Professional's Guide to Licensing and Royalty Agreements. Allworth Press.
4. Warda, M. (2002). How to Register Your Own Copyright. Sphinx Publishing.

Semester: II
2.5. VSC

Course Title	Introduction To Computer Hardware
Course Credits	2
Course Outcomes	After going through the course, learners will be able to
	1. Apply knowledge of computer hardware components to diagnose and solve hardware problems effectively.
	2. Analyse differences between hardware and software, evaluating their roles in computing systems.
	3. Evaluate network types (LAN, MAN, WAN) for advantages and disadvantages in various environments.
	4. Design PC assembly strategies and troubleshooting methodologies to ensure system reliability and performance.
Module 1(Credit 1)	
Learning Outcomes	After learning the module, learners will be able to
	1. Apply knowledge of computer hardware components to propose effective hardware upgrade solutions based on system requirements.
	2. Analyse the characteristics and functionalities of different types of computer memory to optimize performance in computing tasks.
	3. Evaluate the advantages and disadvantages of LAN, MAN, and WAN to design appropriate network configurations for specific organizational needs.
	4. Design strategies for diagnosing and resolving computer hardware issues using systematic diagnostic methods, ensuring efficient problem-solving.
Content Outline	<ul style="list-style-type: none"> • Fundamentals of Computer Hardware: What is Computer Hardware and Hardware Upgrade, Computer Hardware Parts(Components), Hardware vs Software, Hardware Virtualization, Hardware as a service, Computer Hardware Problems and Diagnostic Methods • Computer Memory: Definition, Characteristics of Main Memory, how does Computer Memory work, Types of Computer Memory, Register Memory, Cache Memory, Primary and Secondary Memory and its types, RAM, ROM and Memory units • Computer Network: Basics of Network, LAN, MAN and WAN along with advantages and disadvantages.
Module 2(Credit 1)	
Learning Outcomes	After learning the module, learners will be able to
	1. Apply knowledge of PC assembly by selecting appropriate components, adhering to safety protocols, and executing step-by-step assembly procedures.
	2. Analyse the advantages of different types of computer cables and processors to optimize system performance and connectivity.
	3. Evaluate troubleshooting approaches to diagnose and resolve hardware and software issues in microcomputers and peripheral equipment.

	4. Design effective strategies for managing faulty components, installing I/O devices, and ensuring system stability during boot processes.
Content Outline	<ul style="list-style-type: none"> • PC Assembling and Troubleshooting: • Assembling: <ul style="list-style-type: none"> a) How to build a computer: Choosing the right components, Safety Measures, Steps to build a computer b) Types of Computer Cables and its advantages c) Types of Processors d) Working of Printers and Scanners and its types e) Microcomputers and Motherboards and its types and selection of right motherboard f) Drivers role and types • Troubleshooting: Diagnose and troubleshooting of microcomputer/computer system • hardware & software and other peripheral equipment: Approaches to solve a PC problem, troubleshooting a failed boot before the OS is loaded, Different approaches to installing and supporting I/O devices, Managing Faulty Components.

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1

- Application of theoretical knowledge to practical scenarios.
- Analysis and evaluation of hardware components, memory types, and network configurations.
- Design of effective strategies for hardware troubleshooting.
- Clarity, coherence, and depth of the written report and presentation.

Module 2

- Application of PC assembly knowledge to build a functional custom PC.
- Analysis of different types of computer cables and processors in relation to system performance.
- Evaluation of troubleshooting approaches and effectiveness in resolving hardware and software issues.
- Design of clear and effective strategies for managing faulty components and ensuring system stability.

References

1. Meyers, M. (2017). Introduction to PC Hardware and Troubleshooting. McGraw Hill Education.
2. Lotia, M. (2006). Modern Computer Hardware Course. BPB Publication.
3. Zacker, C., & Rourke, J. (2017). PC Hardware: The Complete Reference. McGraw Hill Education.

Semester: II
2.6 (Swayam/Chetana/MOOC)

Semester: II
2.7 . AEC

Semester: II
2.8 VEC

Semester: II
2.9 CC (Co-Curricular Course-II)