SNDT Women's University, Mumbai

Masters of Computer Applications (MCA) Integrated

Syllabus

As Per NEP – 2020

w.e.f.

A.Y.: 2024-2025

1, Nathibai Thackersey Road, Mumbai-400020

www.sndt.ac.in

Programme Template:

Programme Degree		Masters of Computer Applications (MCA) Integrated
Parenthesis if any (Specialization)		NA
Preamble		The Masters of Computer Applications (MCA) Integrated program is a five-year Postgraduate degree program approved by AICTE as per NEP-2020 program is 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 it's Application. The MCA Integrated 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. During the course of the MCA Integrated program, students are exposed to a wide range of subjects that cover various aspects of computer science Application. These subjects typically include programming languages, data structures, algorithms, computer architecture, operating systems, software engineering, web development, database management systems, computer networks, information security, Artificial intelligence, Machine learning, Deep Learning and IoT. Upon successful completion of the MCA Integrated program, candidates 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. 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.
Programme Specific Outcomes (PSOs)		After completing this programme, learner will
	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		Passed 10+2 examination with Mathematics/ Statistics/ Accountancy as compulsory subjects. Obtained at least 45% marks (40% marks in case of candidates belonging to reserved category) in the above subjects taken together. And Valid MAH-MCA Integrated-2024 CET score. Or As per State CET Cell Norms
Intake (For SN.DT WU 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

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	Fundamental of Computer science	Major (Core)	2	50	0	50
1.3	Open Elective Course-I Digital Marketing	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-curriculum Course-I	CC	2	50	50	0
			22	**550**	**300**	**250**
	*** 1.6, 1.7,1.8,1.9 will be provide by University Basket** *** Co-Curricular Course (Health & Wellness, Yoga education, sports & fitness, Cultural activities, NSS, NCC and Fine/applied/visual/performing arts)**					
	Semester II					
2.1	Object Oriented Programming using C++	Major (Core)	4	100	50	50
2.2	Operating system	Major (Core)	2	50	0	50
2.3	Discrete mathematics	Minor Stream	2	50	0	50
2.4	Open Elective Course-II Intellectual Property Rights	OEC	4	100	50	50
2.5	Open-Source Operating System and Applications	VSC	2	50	0	50
2.6	Vedic mathematics	SEC	2	50	50	0
2.7		AEC	2	50	50	0
2.8		VEC	2	50	0	50
2.9	Co-curriculum Course-II	CC	2	50	50	0
			22	**550**	**250**	**300**
	*** 2.7,2.8,2.9 will be provide by University Basket** *** Co-Curricular Course (Health & Wellness, Yoga education, sports & fitness, Cultural activities, NSS, NCC and Fine/applied/visual/performing arts)**					

Exit with 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	Database Management System	Major (Core)	4	100	50	50
3.3	Mathematics II-Statistical Method	Minor Stream	4	100	50	50
3.4	Open Elective Course-III	OEC	2	50	0	50
3.5	Digital Forensics	VSC	2	50	50	0
3.6		AEC	2	50	0	50
3.7	Field Project	FP	2	50	50	0
3.8	Co-curriculum Course-III University Basket	CC	2	50	50	0
			22	**550**	**300**	**250**
	*** 3.6 will be provide by University Basket** *** 3.7 Field Project will be internal 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 and Microcontroller	Major (Core)	4	100	50	50
4.3	Web 3.0 and Metaverse	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	E- Waste Management	CEP	2	50	50	0
4.8	Co-curriculum Course-IV University Basket	CC	2	50	50	0
			22	**550**	**250**	**300**
	*** 4.6 will be provide by University Basket**					

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

SN	Courses	Type of Course	Credits	Marks	Int	Ext
	Semester V					
5.1	Python Programming	Major (Core)	4	100	50	50
5.2	Software Engineering	Major (Core)	4	100	50	50
5.3	PLSQL and No SQL LAB	Major (Core)-LAB	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 Analytics using Python	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	Computer Network	Major (Core)	4	100	50	50
6.3	Software Testing and Quality Assurance	Major (Core)	2	50	0	50
6.4	Elective – II	Major (Elective)	4	100	50	50
6.5	Artificial Intelligence	Minor Stream	4	100	50	50
6.6	Internship/Apprenticeship	OJT	4	100	50	50
			22	**550**	**250**	**300**

Exit with Degree (3-year)

PG -I

SN	Courses	Type of Course	Credits	Marks	Int	Ext
	Semester VII					
PG1.1	Machine learning	Major (Core)	4	100	50	50
PG1.2	Cloud computing	Major (Core)	4	100	50	50
PG1.3	Data Mining with Analytics	Major (Core)	4	100	50	50
PG1.4	Machine learning lab using Python Lab	Major (Core)	2	50	50	0
PG1.5	Elective-III	Major (Elective)	4	100	50	50
PG1.6	Research Methodology	Minor Stream (RM)	4	100	50	50
			22	**550**	**300**	**250**
	Semester VIII					
PG2.1	Data warehousing	Major (Core)	4	100	50	50
PG2.2	Deep Learning	Major (Core)	4	100	50	50
PG2.3	Fuzzy logic	Major (Core)	4	100	50	50
PG2.4	DL Lab using R Programming	Major (Core)	2	50	0	50
PG2.5	Elective-IV	Major (Elective)	4	100	50	50
PG2.6	OJT	OJT	4	100	50	50
			22	**550**	**250**	**300**

PG –II

SN	Courses	Type of Course	Credits	Marks	Int	Ext
	Semester IX					
PG3.1	Managerial Economics	Major (Core)	4	100	50	50
PG3.2	Big Data Analytics	Major (Core)	4	100	50	50
PG3.3	Cyber Security	Major (Core)	4	100	50	50
PG3.4	Distributed Computing	Major (Core)	2	50	0	50
PG3.5	Elective-V	Major (Elective)	4	100	50	50
PG3.6	RP-I	RP	4	100	50	50
			22	**550**	**250**	**300**
	Semester X					
PG4.1	NLP	Major (Core)	4	100	50	50
PG4.2	Mobile Computing	Major (Core)	4	100	50	50
PG4.3	NLP Lab	Major (Core)	2	50	0	50
PG4.4	Elective-VI	Major (Elective)	4	100	50	50
PG4.5	RP-II/OJT	RP	8	200	150	50
			22	**550**	**300**	**250**

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**
	• Apply logic to create programs in C.
	• Analyze and understand computer programming language concepts.
	• Evaluate and interpret the use of pointers, including their declarations, initialization, and operations.
	• 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**
	• Learn steps in problem solving using C.
	• Understand structure, keywords, operators, and functions of C programming.
	• Evaluate the concept of I/O functions, header files, and preprocessor directives.
	• Design and apply concepts of the C language.
Content Outline	• **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.LanguageFundamentals:Characterset,CTokens,Keywords, Identifiers,Variables, Constant, Data Types, Comments

	• **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**
	• Gain proficiency in writing C programs to solve various problems.
	• Learn the syntax and semantics of the C language, including its specific features such as pointers and memory management.
	• Analyze the difference between structure and union.
	• Design and handle operations of the files.
Content Outline	• **Control structures** ➢ 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**
	• Apply algorithms by writing C code.
	• Analyze and trace the execution of C programs.
	• Evaluate the use of pointers, arrays, and the pre-processor in programs.
	• 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**
	• Apply operators to write simple programs.
	• Analyze and use control, iterative, and jumping statements.

	• Evaluate the use of break and continue statements.
	• Design programs with header files and preprocessor directives.
Content Outline	• **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**
	• Apply functions in programs.
	• Analyze the declaration, initialization of pointers, and passing arrays to functions.
	• Evaluate the definitions and declarations of structure variables in programs.
	• Design programs effectively using functions, pointers, and structures.
Content Outline	• **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	**FUNDAMENTALS OF COMPUTER SCIENCE**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to**
	• Apply computer fundamentals comprehensively.
	• Analyze and solve problems using programming and software development skills.
	• Evaluate proficiency in programming languages, software development life cycles, and debugging techniques.
	• Design software with strong internet literacy and effective problem-solving skills.
Module1 (Credit1)	
Learning Outcomes	**After learning the module, learners will be able to**
	• Apply knowledge of essential computer components, including CPU, input/output devices, memory, and storage.
	• Analyze internal data representation, including number systems, binary arithmetic, Boolean algebra, and logic gates.
	• Evaluate the functions of various input/output devices and interfaces within computer systems.
	• Design an understanding of basic computer organization and the roles of CPU, memory, and secondary storage.
Content Outline	• **Knowing computer:** What is Computer, Basic Applications of Computer; Components of Computer System, Central Processing Unit (CPU), Input/output Devices, Computer Memory, Concepts of Hardware and Software; Concept of Computing, Data and Information, classification of computers, various generations of computers; What is an Operating System; Different Popular Operating Systems; The User Interface, System Software: System software Vs. Application Software, Types of System Software, Introduction and Types of Operating Systems • **Internal data Representation in Computers and Digital System Design:** Number Systems Used in Computers, Converting from One Number System to Another, Binary Arithmetic, Boolean Algebra, Boolean Functions, Logic Gates, Logic Circuits • **Computer Architecture:** Basic Functions of a Computer, Basic Computer Organization, CPU Architectures, Memory Architectures, Secondary Storage, Classification of Secondary Storage, Memory Storage Devices, Input- Output Devices ,Input Devices, Output Devices, I/O Interfaces.

Module2 (Credit1)	
Learning Outcomes	**After learning the module, learners will be able to**
	• Apply knowledge of computer language evolution and types.
	• Analyze programming constructs, algorithms, and flowcharting skills.
	• Evaluate the roles of language processors, software development, and the SDLC.
	• Design an understanding of basic computer networks, internet connectivity, and the architecture of the World Wide Web.
Content Outline	• **Computer Languages:** Definition, Generations of computer languages, Types of Languages, Language Processors: Assembler, Interpreter, Compiler, Linker and Loader. Programming constructs, Algorithm & flowchart. • **Software:** Basic Concepts and Terminologies, What is Software? ,Relationship between Hardware and Software, Software Development Life Cycle (SDLC), Advantages of SDLC Model, Limitations of SDLC Model, Software Testing and Debugging • **Introduction to Internet:** WWW and Web Browsers: Basic of Computer networks; LAN, WAN; Concept of Internet; Applications of Internet; connecting to internet; What is ISP; Knowing the Internet; Basics of internet connectivity related troubleshooting, World Wide Web; Web Browsing software, Search Engines; Understanding URL; Domain name; IP Address; Using e-governance website

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1 :

Write an essay discussing the fundamental components and concepts of computer systems.

1. Explain the basic components of a computer system, including CPU, input/output devices, memory, and storage.
2. Describe the internal data representation in computers, covering number systems, binary arithmetic, Boolean algebra, and logic gates.
3. Discuss computer architecture, including CPU architectures, memory architectures, and secondary storage.
4. Analyze the functions and roles of input/output devices and interfaces within computer systems.
5. Provide examples and illustrations where necessary to clarify concepts.

Module 2 :

Develop a presentation on computer languages, software development, and the basics of internet connectivity.

1. Explain the evolution and types of computer languages, including the generations of

languages and the role of language processors (assembler, interpreter, compiler, linker, and loader).

2. Discuss programming constructs, algorithms, and flowcharting skills.
3. Describe basic concepts and terminologies related to software, including the software development life cycle (SDLC), software testing, and debugging.
4. Provide an introduction to the internet, covering basic computer network concepts (LAN, WAN), internet connectivity, and the architecture of the World Wide Web.
5. Include examples and case studies to illustrate key points effectively.

References: -

1. Sinha, P. K., & Sinha, P. (2004). Computer Fundamentals. BPB Publications.
2. Srivastava, C. (2010). Fundamentals of Information Technology. Kalyani Publishers.
3. Rajaraman, V. (2003). Fundamentals of Computers (4th ed.). PHI Publication.
4. Jain, R. K. (2010). Fundamentals of Programming. S. Chand Publication.

Semester: I

1.3 Open Elective Course-I (OEC-I)

Course Title	**DIGITAL MARKETING**
Course Credits	**4**
Course Outcomes	**After completion of this Course, the students will be able to**
	• Apply principles and foundations of digital marketing.
	• Analyze data to optimize marketing strategies and campaigns.
	• Evaluate digital marketing performance using analytics tools.
	• Design effective marketing using digital channels, customer insights, and segmentation.
Module1 (Credit1)	
Learning Outcomes	**After learning this module, learners will be able to**
	• Apply principles and techniques of digital marketing using various channels and tools.
	• Analyze and understand SEO techniques.
	• Evaluate the effectiveness of digital marketing strategies.
	• Design comprehensive digital marketing campaigns.
Content Outline	• **Basics of Websites & Digital Marketing: -** ➢ 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: -** ➢ 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	**After learning this module, learners will be able to**
	• Apply strategies for effective campaigns.
	• Analyze website metrics using Google Analytics.
	• Evaluate important metrics for SEM campaigns.
	• Design and optimize campaigns based on analytics data.
Content Outline	• **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	**After learning this module, learners will be able to**
	• Apply business analytics techniques for data-driven marketing decisions.

	• Analyze data insights to optimize marketing strategies and campaigns.
	• Evaluate the effectiveness of data-driven marketing decisions.
	• Design marketing strategies based on interpreted data insights.
Content Outline	• **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**
	• Apply appropriate metrics and analytics tools to measure digital marketing campaign performance.
	• Analyze data gathered from campaigns to evaluate their effectiveness.
	• Evaluate the performance of digital marketing campaigns based on measured metrics.
	• Design strategies for optimizing digital marketing campaign performance based on evaluation results.
Content Outcomes	• **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**
	• Apply fundamental concepts of web technology, including the World Wide Web, HTTP protocol, web browsers, and web servers.
	• Analyze HTML and CSS proficiency to create visually appealing web pages.
	• Evaluate the use of JavaScript to enhance interactivity and add dynamic functionality to web pages.
	• 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**
	• Apply deep understanding of web technology concepts, protocols, and standards.
	• Analyze HTML and CSS proficiency to create visually appealing web pages.
	• Evaluate interactive and dynamic web page development using JavaScript and DOM manipulation.
	• Design and implement server-side scripts for handling user requests and generating dynamic web content.
Content Outline	• **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**
	• Apply integration of databases with web applications for efficient data handling.
	• Analyze and implement secure coding practices to safeguard web applications.
	• Evaluate the effectiveness of web development frameworks and tools in streamlining processes.
	• Design database integration methods for optimizing data storage and retrieval in web applications.
Content Outline	• 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.

Course Syllabus

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**
	• Apply object-oriented programming concepts in C++.
	• Analyze problems and develop C++ programs to solve them.
	• Evaluate the use of file input/output in C++.
	• Design solutions using object-oriented programming principles in C++.
Module1 (Credit1)	
Learning Outcomes	**After learning the module, learners will be able to**
	• Apply fundamental programming concepts including variables, data types, control structures, functions, arrays, and objects.
	• Analyze and implement object-oriented programming concepts like objects, classes, and defining functions and variables.
	• Evaluate the use of object-oriented programming in solving programming problems.
	• Design solutions using a combination of fundamental programming and object-oriented concepts.
Content Outline	• **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**
	• Apply concepts of constructors, inheritance (including its types), and polymorphism in C++.
	• Analyze file input/output handling in C++ and class templates.
	• Evaluate the effectiveness of constructors, inheritance, and polymorphism in solving programming tasks.
	• Design solutions involving file input/output operations and the utilization of class templates in C++.
Content Outline	• **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**
	• Apply object-oriented programming concepts by creating programs with classes and objects in C++.
	• Analyze the implementation of object-oriented programming concepts in C++.
	• Evaluate the effectiveness of stream I/O and file I/O in developing applications.
	• 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**
	• Apply the learned concepts by writing simple programs using classes and objects in C++.
	• Analyze the implementation of object-oriented programming concepts in C++ to understand their functionality and usage.
	• Evaluate the effectiveness of object-oriented programming in solving programming tasks.
	• Design solutions using object-oriented programming concepts to improve code structure and modularity in C++.
Content Outline	• **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**
	• Apply the concepts of inheritance, constructors, and operator overloading by writing simple programs in C++.
	• Analyze the process of reading and creating text files using C++ programs.
	• Evaluate the effectiveness of inheritance, constructors, and operator overloading in solving programming tasks.
	• Design programs that utilize inheritance, constructors, and operator overloading for improved code organization and functionality in C++.

Content Outline	• 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	**OPERATING SYSTEMS**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to**
	• Apply knowledge of computer system organization and architecture to solve complex problems.
	• Analyze the structure, operations, and design principles of operating systems.
	• Evaluate virtualization techniques, including virtual machines and operating system generation, for efficiency and effectiveness.
	• Design processor management and scheduling strategies for multi-processor systems to optimize performance.
Module1 (Credit1)	
Learning Outcomes	**After learning this Module, Learners will be able to**
	• Apply knowledge of computer system organization and architecture, with a focus on the role of operating systems.
	• Analyze the structure and operations of operating systems, including key components like process management, memory management, storage management, protection, and security.
	• Evaluate the various operating system services available to users and applications for efficiency and effectiveness.
	• Design understanding of the user-operating system interface, system calls, and types of system programs to enhance system functionality and usability.
Content Outline	• **Introduction to Operating Systems (OS):** Computer-System Organization, Computer-System Architecture, Operating-System Structure, Operating-System Operations, Process Management, Memory Management, Storage Management, Protection and Security, Distributed Systems, Special-Purpose Systems, Computing Environments. • Operating-System Services, User Operating-System Interface, System Calls, Types of System Calls, System Programs, Operating-System Design and Implementation, Operating-System Structure, Virtual Machines, Operating-System Generation. • **Processor Management:** Process concept, Process scheduling, Operations on Processes, Interprocess Communication, Multithreading models, threading issues, Process scheduling algorithms, Thread scheduling, Multiple processor Scheduling. • **Process Coordination:** Synchronization, Semaphores, Monitors, Deadlocks characterization, Methods for handling deadlocks, Deadlock prevention, Deadlock Avoidance, Deadlock detection, recovery from deadlock.
Module2 (Credit 1)	
Learning Outcomes	**After learning this Module, Learners will be able to**
	• Apply various memory management techniques such as swapping, contiguous memory allocation, paging, and segmentation in practical scenarios.
	• Analyze the structure and function of the page table to understand its role in memory addressing.

	• Evaluate fundamental file concepts, including file access methods, directory structures, file sharing, and protection mechanisms, to ensure efficient file management.
	• Design efficient file systems by understanding their organization, structure, directory implementation, allocation methods, and free-space management to optimize storage and retrieval operations.
Content Outline	• **Memory Management:** Swapping, Contiguous Memory Allocation, Paging, Structure of the Page Table, Segmentation • **Virtual memory Management:** Demand Paging, Copy- on-Write, Page replacement, Allocation of Frames, Thrashing. • **File Management:** File Concept, File Access Methods, Directory Structure, File Sharing, File Protection, File-System Structure, File- System Implementation, Directory Implementation, Allocation Methods, Free-Space Management, Efficiency and Performance, Recovery, Log-Structured File Systems, NFS.

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1:
Design a presentation that explains the fundamental concepts of operating systems and their key components.

1. Create slides that cover topics such as computer system organization, architecture, operating system structure, and operations.
2. Explain the role of operating systems in process management, memory management, storage management, protection, and security.
3. Describe the various operating system services available to users and applications, highlighting their efficiency and effectiveness.
4. Discuss the user-operating system interface, system calls, and types of system programs to enhance system functionality and usability.
5. Include examples and illustrations to clarify complex concepts and engage the audience effectively.

Module 2:
Develop a research paper that explores memory management techniques and file system concepts in operating systems.

1. Discuss various memory management techniques such as swapping, contiguous memory allocation, paging, and segmentation, explaining their advantages, disadvantages, and practical applications.
2. Analyze the structure and function of the page table in virtual memory management, highlighting its role in memory addressing and address translation.
3. Evaluate fundamental file concepts, including file access methods, directory structures, file sharing, and protection mechanisms, to ensure efficient file management.
4. Design efficient file systems by understanding their organization, structure, directory implementation, allocation methods, and free-space management to optimize storage and retrieval operations.
5. Provide real-world examples and case studies to illustrate the implementation and impact of memory management and file system concepts in operating systems.
6. Include references to relevant literature and research papers to support your analysis and findings.

References/Text Books:-

1. Silberschatz, A., Galvin, P. B., & Gagne, G. (2005). Operating System Concepts (7th ed.). John Wiley & Sons, Inc.

2. Milenkovic, M. (2001). Operating Systems Concepts And Design (2nd ed.). McGraw-Hill

International Editions.

3. Stallings, W. (2005). Operating Systems: Internals and Design Principles (5th ed.). Prentice Hall.
4. Tanenbaum, A. (2007). Modern Operating Systems (3rd ed.). Pearson Education.

Semester: II

2.3 Minor Stream

Course Title	**DISCRETE MATHEMATICS**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to**
	• Apply sets, relations, functions, and mathematical reasoning techniques like the Binomial Theorem and Induction to solve discrete mathematical problems.
	• Analyze and investigate a variety of discrete mathematical problems using acquired knowledge and skills.
	• Evaluate the effectiveness of sets, relations, functions, and mathematical reasoning techniques in solving discrete mathematical problems.
	• Design solutions for discrete mathematical problems by applying sets, relations, functions, and mathematical reasoning techniques effectively.
Module1 (Credit1)	
Learning Outcomes	**After learning the module, learners will be able to**
	• Apply algorithms to solve problems related to GCD, LCM, and various theorems.
	• Analyze and solve problems involving sets, relations, and functions.
	• Evaluate solutions to basic problems using the Binomial Theorem and Mathematical Induction.
	• Design problem-solving strategies for GCD, LCM, theorems, sets, relations, functions, the Binomial Theorem, and Mathematical Induction to address various mathematical challenges effectively.
Content Outline	• **Properties of Integers:** Definition of GCD, LCM, Theorems Euclidean algorithm and problems
	➢ **Set Theory**: Definition of Sets, Subsets, Cardinality of Sets, Types of sets: Equal Sets, Universal Sets, Finite and Infinite Sets, proper set, power sets, Operations on Sets: Union, Intersection, Complement of Sets, set difference, Cartesian Product, Venn Diagrams, and Algebra of sets
	➢ **Relations:** Definitions of Relation, Reflexive Relation, Symmetric Relation, Transitive relation, Equivalence Relation Recurrence Relation: Definitions and problems
	➢ **Functions**: Define Function, Injective Functions, Surjective Functions, Bijective Functions, Composite Function, Inverse of a Function, Domain and Range
	➢ **Binomial Theorem and Mathematics Induction:** Binomial Theorem: Statement and basic problems and Mathematical Induction: Principles and basic problems.

Module2 (Credit 1)	
Learning Outcomes	**After learning the module, learners will be able to**
	• Apply permutations and combinations concepts to solve mathematical problems.
	• Analyze and solve problems involving matrices and determinants.
	• Evaluate solutions to permutations, combinations, matrices, and determinants problems.
	• Design problem-solving strategies for permutations, combinations, matrices, and determinants to address various mathematical challenges effectively.
Content Outline	➢ **Permutations and Combinations:** Permutation: Definition, Basic Permutation, Permutation with Repetition and Circular Permutation and basic problems, Combination: Definition and basic problems ➢ **Matrices and Determinants:** Matrices: Definition, Operations on Matrices, Square Matrix and its Inverse, Solution of Equations using Matrices, Determinants: Properties of Determinants, Solution of equations using Determinants

Assignments/Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1:
Develop a set of problems and solutions focusing on the topics covered in Module 1.

1. Create problems related to GCD, LCM, Euclidean algorithm, sets, relations, functions, recurrence relations, the Binomial Theorem, and Mathematical Induction.
2. Each problem should require students to apply the concepts learned in the module to find solutions.
3. Provide detailed solutions for each problem, including step-by-step explanations and any relevant theorems or algorithms used.
4. Include a variety of difficulty levels to challenge students and reinforce their understanding of the material.
5. Organize the problems and solutions into a document or presentation for distribution to students.

Module 2:
Design a series of mathematical problems focusing on permutations, combinations, matrices, and determinants.

1. Create problems related to permutations, combinations (with and without repetition), circular permutations, matrix operations, inverses, solutions of equations using matrices, and properties of determinants.
2. Each problem should require students to apply the concepts learned in Module 2 to find solutions.
3. Provide detailed solutions for each problem, demonstrating the step-by-step process and any relevant formulas or techniques used.
4. Include a mix of theoretical and practical problems to ensure students have a comprehensive understanding of the material.
5. Organize the problems and solutions into a document or presentation for distribution to students.

References: -

1. Kolman, B., Busby, R. C., & Ross, S. (Year). Discrete Mathematical Structures and Graph Theory.

2. Doerr, A., & Levasseur, K. (1988). Applied Discrete Structures for Computer Science. Galgotia Publications.

3. Trembley, J. P., & Manohar, R. (1987). Discrete Mathematical Structures with Applications to Computer Science. McGraw-Hill.

4. Chakraborty, S. K., & Sarkar, B. K. (2011). Discrete Mathematics. Oxford Higher Education.

5. Liu, C. L., & Mohapatra, D. P. (2008). Elements of Discrete Mathematics: A Computer Oriented Approach (3rd ed.). Tata McGraw-Hill.

Semester: II

2.4 Open Elective Course-II (OEC-II)

Course Title	**INTELLECTUAL PROPERTY RIGHTS**
Course Credits	**4**
Course Outcomes	**After going through the course, learners will be able to**
	• Apply knowledge of Intellectual Property Rights to protect creative work effectively.
	• Analyze the use of Intellectual Property in various contexts.
	• Evaluate the effectiveness of using Intellectual Property to safeguard creative work.
	• Design strategies for the registration process of Copyright, Patent, and Trademark to ensure legal protection of creative work.
Module1 (Credit1)	
Learning Outcomes	**After learning the module, learners will be able to**
	• Apply knowledge and understanding of the justifications and rationales for protecting intellectual property in practical scenarios.
	• Analyze the reasons behind the need to protect intellectual property rights.
	• Evaluate the effectiveness of intellectual property protection in promoting innovation and creativity.
	• Design strategies to advocate for and support the protection of intellectual property rights based on understanding the justifications and rationales.
Content Outline	• **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**
	• Apply knowledge and understanding of different countries' Intellectual Property Rights (IPR) acts in legal contexts.
	• Analyze the similarities and differences between various countries' IPR acts.
	• Evaluate the effectiveness of different countries' IPR acts in protecting intellectual property.
	• Design strategies for navigating and complying with different countries' IPR acts based on understanding and knowledge.

Content Outline	• **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	**After learning the module, learners will be able to**
	• 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	• **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	**After learning the module, learners will be able to**
	• 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	• **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	**OPEN SOURCE OPERATING SYSTEM AND APPLICATIONS**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to**
	• 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	**After learning the module, learners will be able to**
	• 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	• **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	**After learning the module, learners will be able to**
	• 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	• **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

2.6 SEC

Course Title	**VEDIC MATHEMATICS**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to**
	• Apply Ancient Vedic Mathematics techniques to solve mathematical problems efficiently.
	• Analyze the principles behind Faster Calculation Methods for improved computation speed.
	• Evaluate the effectiveness of Ancient Vedic Mathematics techniques and Faster Calculation Methods in various problem-solving scenarios.
	• Design strategies for learning and practicing Ancient Vedic Mathematics techniques and Faster Calculation Methods to enhance mathematical proficiency.
Module1 (Credit1)	
Learning Outcomes	**After learning the Module, learners will be able to**
	• Apply the understanding of the differences between general mathematics and Vedic mathematics in problem-solving.
	• Analyze techniques for rapidly adding single, double, and triple digits using Vedic mathematics methods.
	• Evaluate the efficiency of Vedic mathematics in performing rapid addition tasks compared to traditional methods.
	• Design practice exercises to learn and master rapid addition techniques for single, double, and triple digits using Vedic mathematics principles.
Content Outline	• Introduction of Vedic Maths • Benefits of Vedic Maths • Difference between general Maths and Vedic Maths • Mental Maths Addition • Rapid Addition-Single to Double-Digit • Rapid Addition-Double to Double-Digit • Rapid Addition-Triple to Triple-Digit • Left to Right Addition
Module2 (Credit1)	
Learning Outcomes	**After learning the Module, learners will be able to**
	• Apply multiplication techniques to solve mathematical problems efficiently.
	• Analyze various multiplication tricks to understand their underlying principles.
	• Evaluate the effectiveness of different multiplication tricks in solving multiplication problems quickly.
	• Design practice exercises to master different multiplication tricks and enhance multiplication skills.
Content Outline	• Multiplication with Double Digit to Single Digit numbers • Multiplication by Multiples of 10 • Traditional Multiplication • Multiplication with Tricks

Assignments/Activities to wards Comprehensive Continuous Evaluation (CCE)

Module 1 :
Create a series of practice exercises and a summary report on the benefits and differences between Vedic mathematics and traditional mathematics in rapid addition.

1. Develop a set of practice exercises for rapid addition using Vedic mathematics principles, covering single-digit, double-digit, and triple-digit addition problems.
2. Provide step-by-step explanations for each rapid addition technique, including mental maths addition and left-to-right addition.
3. Analyze and compare the efficiency of Vedic mathematics rapid addition techniques with traditional methods, highlighting the differences and benefits of Vedic maths.
4. Include examples and case studies to illustrate the application of Vedic mathematics in real-world scenarios.
5. Design a summary report summarizing the key concepts, benefits, and differences between general mathematics and Vedic mathematics in problem-solving.

Module 2:
Develop a tutorial and practice exercises focusing on various multiplication techniques derived from Vedic mathematics principles.

1. Create a tutorial explaining multiplication techniques for multiplying double-digit numbers with single-digit numbers, multiplying by multiples of 10, and traditional multiplication methods.
2. Include step-by-step instructions for each multiplication technique, demonstrating how to apply Vedic mathematics principles to solve multiplication problems efficiently.
3. Analyze the effectiveness of different multiplication tricks in terms of speed and accuracy, comparing them to traditional multiplication methods.
4. Develop a series of practice exercises covering different multiplication techniques, allowing learners to practice and master each method.
5. Provide solutions and explanations for the practice exercises, ensuring learners understand the underlying principles behind each multiplication trick.

References:

1. Tirtha, B. K. (2012). Mental Calculation. Sheth Publishing House.

2. Tirtha, B. K. (2013). Vedic Mathematics. Motilal Banarsidass Publishers.