Punyashlok Ahilyadevi Holkar Solapur University, Solapur

Name of the Faculty: Science and Technology

NEP-2020

Syllabus: - Entire Computer Science

Name of the Program: B.Sc. I (ECS) (Sem.– I & II) (NEP-2020)

(Syllabus to be implemented w.e.f. June 2024)

Punyashlok Ahilyadevi Holkar Solapur University, Solapur

B. Sc. (Entire Computer Science)- I

Preamble: B. Sc. (Entire Computer Science) is multiple entries and multiple exit option 4- year programme specializing in computer science, software and hardware-related aspects. B.Sc. (ECS) programme is perfect for students who want to make a career in computers. Major subjects in this programme include Computer Programming theory, Mathematical foundation, Advanced Programming using Python, machine learning, Java, etc. The course curriculum is inclusive of theory and practical which makes the students well trained and skillful in programming, software, and networks.

The objective of the Programme:

- To develop problem-solving abilities using a computer.
- To build the necessary skill set and analytical abilities for developing computer-basedsolutions for real-life problems.
- To train students in professional skills related to Software Industry.
- To prepare the necessary knowledge base for research and development in Computer Science.
- To help students build up a successful career in Computer Science and to produce entrepreneurs who can innovate and develop software products. Programme Outcome: B.Sc. (ECS) programme has been designed to prepare graduates for attaining the following specific outcomes:
- An ability to apply knowledge of mathematics, statistics and computer science in practice.
- An ability to enhance a comprehensive understanding of the theory and its applicationin diverse fields.
- The program prepares the young professional for a range of computer applications, computer organization, techniques of Computer Networking, Software Engineering, Web Development, Database management and advanced Java
- An ability to design a computing system to meet desired needs within realistic constraints such as safety, security and applicability in multidisciplinary teams with a positive attitude.
- To enhance the programming skills of young IT professionals, the program hasintroduced the concept of project development in each language/technology learned during the curriculum.

Eligibility for B.Sc. (ECS) Part-I:

The candidate passing the Higher Secondary Examination Conducted by the Maharashtra State Board of Higher Secondary Education, with Science stream, MCVC with Science Subjects, D. Pharm, Diploma, Engineering, Agricultural Diploma, Diary Diploma shall be allowed to enter upon the B. Sc. Part-I Course.

OR

An examination of any other statutory University or an Examining Body is recognized as equivalent thereto.

Repeater Students will be allowed to take fresh admission to the same class with the same subjects or different subjects.

Programme Learning Outcomes:

These outcomes describe what students are expected to know and can do by the time of graduation. They relate to the skills, knowledge, and behaviour that students acquire in their graduation through the program

Programme Learning Outcomes for BSc (Entire Computer Science):

The Bachelor of Science (Entire Computer Science) programme enables students to attain, bythe time of graduation:

PLO-1. Demonstrate an aptitude for Computer Programming and Computer-based problem-solving skills.

PLO-2. Display the knowledge of appropriate theory, practices and tools for the specification,design, implementation

PLO-3. Ability to learn and acquire knowledge through online courses available at differentMOOC Providers.

PLO-4. Ability to link knowledge of Computer Science with other two chosen auxiliarydisciplines of study.

PLO-5. Display an ethical code of conduct in the usage of the Internet and Cybersystems.

PLO-6. Ability to pursue higher studies of specialization and to take up technicalemployment.

PLO-7. Ability to formulate, model, design solutions, procedures and use software tools tosolve real-world problems and evaluate.

PLO-8. Ability to operate, manage, deploy, and configure computer network, hardware, andsoftware operation of an organization.

PLO-9. Ability to present results using different presentation tools.

PLO-10. Ability to appreciate emerging technologies and tools.

PLO-11. Apply standard Software Engineering practices and strategies in real-time softwareproject development19

PLO-12. Design and develop computer programs/computer-based systems in the areasrelated to algorithms, networking, web design, cloud computing, IoT and data analytics.

PLO-13. Acquaint with the contemporary trends in industrial/research settings and therebyinnovate novel solutions to existing problems

PLO-14. The ability to apply the knowledge and understanding noted above to the analysisof a given information-handling problem.

PLO-15. The ability to work independently on a substantial software project and as aneffective team member.

Subject/ Core Course	Name and Type of the Paper		Hrs./week			Total Marks Per Paper	UA	CA	Credits
	Type	Name	L	T	P				
Sem-I									
Major **(Select any three)**	DSC1-1	OOP'S with C++-I	2	-	-	50	30	20	2
	Practical	Practical based on DSC1-1	-	-	4	50	30	20	2
	DSC2-1	Python Programming-I	2	-	-	50	30	20	2
	Practical	Practical based on DSC2-1	-	-	4	50	30	20	2
	DSC3-1	Fundamental of Computers	2	-	-	50	30	20	2
	Practical	Practical based on DSC3-1	-	-	4	50	30	20	2
	DSC4-1	Software Engineering	2	-	-	50	30	20	2
	Practical	Practical based on DSC4-1	-	-	4	50	30	20	2
	DSC5-1	Digital System	2	-	-	50	30	20	2
	Practical	Practical based on DSC5-1	-	-	4	50	30	20	2
SEC/ VSC	SEC-1	Introduction to Web Design	2			50	30	20	2
AES, IKS, VEC	L1-1	English	2	-	-	50	30	20	2
	IKS	To be selected from the Basket of IKS	2	-	-	50	30	20	2
	VEC-1	Constitution of India	2	-	-	50	30	20	2
CC1	CC1	Community Engagement & Services	2	-	-	50	30	20	2
Total			**16**		**12**	**550**	**330**	**220**	**22**
Sem-II									
Major **(Select any three)**	DSC1-2	OOP'S with C++- II	2	-	-	50	30	20	2
	Practical	Practical based on DSC1-2	-	-	4	50	30	20	2
	DSC2-2	Python Programming-II	2	-	-	50	30	20	2
	Practical	Practical based on DSC2-2	-	-	4	50	30	20	2
	DSC3-2	Basics of Operating System	2	-	-	50	30	20	2
	Practical	Practical based on DSC3-2	-	-	4	50	30	20	2
	DSC4-2	Linux and Shell Programming	2	-	-	50	30	20	2
	Practical	Practical based on DSC4-2	-	-	4	50	30	20	2
	DSC5-2	Computational Mathematics	2	-	-	50	30	20	2
	Practical	Practical based on DSC5-2	-	-	4	50	30	20	2
Generic/ Open Elective Courses	GE1/ OE1	Office Automation	2	-	-	50	30	20	2
SEC/ VSC	SEC2	Advanced Web Designing	2			50	30	20	2
AES, IKS, VEC	L1-2	English	2	-	-	50	30	20	2
	VEC-2	Environmental science	2	-	-	50	30	20	2
CC2	CC2	Community Engagement & Services	2	-	-	50	30	20	2
Total			**16**	**-**	**12**	**550**	**330**	**220**	**22**
Grand Total			**32**	**-**	**24**	**1100**	**660**	**440**	**44**

Abbreviations:				
L: Lectures	**T**: Tutorials	**P**: Practical	**UA** : University Assessment	**CA** : College Assessment
Generic/ Open Electives: **GE/OE**			Skill Enhancement Courses: **SEC**	
Indian Knowledge System: **IKS**			Ability Enhancement Courses: **AES**	
Value Education Courses: **VEC**			Vocational Skill and Skill Enhancement Courses: **VSEC**	
Co-curricular Courses: **CC**				

Student contact hours per week: 24 Hours (Min.)

Total Credits for B.Sc[ECS]-I (Semester I and II): 44

Medium of instruction: English

I. Practical Examination is the Semester wise after theory Examination.

II. Duration of Practical Examination as per respective BOS guidelines.

III. Separate passing is mandatory for Theory, Internal and Practical Examination.

Exit Option at Level 4.5:

Students can exit after Level 4.5 with under certificate course in ComputerProgramming if he/she complete the courses equivalent to a minimum of 44 credits and an additional.
4 credits core NSQF course/Internship.

Course Structure:

Lectures and Practicals should be conducted as per the scheme of lectures and practicals indicated inthe course structure.

Teaching and Practical Scheme

I. Contact session for teaching 60 minutes each.

II. One Practical Batch should be of 20 students.

Assessment

I. The final practical examination will be conducted by the University appointed examiners internalas well as external at the end of the semester for each lab course and marks will be submitted to the university by the panel.

II. The practical examination will be conducted semester-wise to maintain the relevance of the respective theory course with the laboratory course.

III. The final examinations shall be conducted at the end of the semester.

Practical Examination:

I. Each paper carries 30 Marks.

II. ***Duration of Practical Examination:*** 2 Hrs.

III. ***Nature of Question Paper:*** There will be four questions of 10 marks each. Students will attemptany two out of four questions.

IV. Certified Journal carries 5 Marks and Viva voce carries 5 Marks.

Standard of Passing:

I. Minimum 12 marks in each subject. There shall be separate passing for theory (semester endexam and Internal) and practical also.

II. Admission to B.Sc. (Entire Computer Science) Part II is allowed even if the student fails in allthe subjects of part I.

III. Admission to B.Sc. (Entire Computer Science) Part III is allowed only if a student has passed onall the subjects of B.Sc. (Entire Computer Science) Part I.

Board of Paper Setters /Examiners:

For each semester-end examination, there will be a board of Paper setters and examiners for every course. While appointing paper setters/examiners, care should be taken to see that there is at least one person specialized in each unit of the course.

Credit system implementation:

As per the University norms.

Fees Structure:

As approved by the PAHS University fee fixation committee.

Intake Capacity: 80

Award of Class:

Grading: PAHS University has introduced a ten-point grading system as follows:

Sr. No.	Grade Abbreviation	From(Marks)	To (Marks)	Status	Grade Point	Description
1.	O	80	100.00	Pass	10.00	Excellent / Outstanding
2.	A^+	70	79.99	Pass	9.00	Very Good
3.	A	60	69.99	Pass	8.00	Good
4.	B+	55	59.99	Pass	7.00	Fair
5.	B	50	54.99	Pass	6.00	Above Average
6.	C+	45	49.99	Pass	5.00	Average
7.	C	40	44.99	Pass	4.00	Below Average
8.	F	0	39.99	Fail	0.00	Fail

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
Sem:	I				
Paper Category:	DSC1-1 **(Major)**				
Paper Name:	OOP'S with C++-I				
Credit:	02		**Theory:**	2 Hrs./Week	
Marks:	**UA:**	30	**CA:**	20	**Total:** 50

Course Objective:

Students will learn;

1. The basic programming and OOPs concepts
2. Creating C++ programs
3. Tokens, expressions and control structures in C++
4. Arranging same data systematically with arrays
5. Classes and objects in C++

Course Outcomes:

Upon successful completion of this course, students will be able to

1. Describe OOPs concepts
2. Use functions and pointers in your C++ program
3. Understand tokens, expressions, and control structures
4. Explain arrays and strings and create programs using them

Unit-I: Introduction to (Object Oriented Programming) OOP: [15]

Introduction to algorithm and flowchart, Introduction to OOP, Featuresof OOP's- Class, Object, Data Abstraction and encapsulation, Data hiding, Message passing, polymorphism, inheritance, delegation, Comparison between POP(Procedural Oriented Programming) and OOP, Advantages of OOP's, Application of OOP

Introduction to C++: History of C++, C++ basics(C++ tokens)- Keywords, identifiers, data types, constants, operators, special symbols, control flow statements- Decision and iterative statements, Types of Variables- Value, pointer and reference, Structure of C++ program, Basics Input and Output- cin, cout objects, Introduction to array, pointer and template, Function and its types, Default argument, Parameter passing methods, inlinefunction

Unit-II: Classes and Objects: [15]

Introduction to class and object, Defining class (class specification), Creating object, Access specifier (Visibility modes)-public, protected, private, Class members- data members, member function, Defining member function inside and outside the class, Static data members, static member functions, Pointer to object, Array of object, Returning objects from functions, Passing object as parameter by value, by pointer, by reference, Dynamic memory allocation (new, delete), Friend function and friend class, nesting of classes, Constructors, characteristics of constructor, Types of constructor- default, parameterized and copy Constructor overloading, Constructor with default argument, Destructor, characteristics of destructor, Static polymorphism (Function and Operator overloading) , rules to overload operator, unary and binary operator overloading, overloading operator using member function and friend function, Type conversion (type casting)- implicit and explicit.

Reference books:

1. OOP in C++ – E-balagurusamy
2. Mastering C++-K. R. Venugopal
3. The Complete Reference C++-Herbert Schildt

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
Sem:	I				
Paper Category:	Practical **(Major)**				
Paper Name:	Practical based on DSC1-1				
Credit:	02	**Practical:**	4 Hrs./Week		
Marks:	**UA:** 30	**CA:** 20	**Total:** 50		

1. Write different programs in 'C++' language that shows use of array, pointers variable,reference variable, cin and cout objects, scope resolution operators, basic operators
2. Write a program that shows parameter passing techniques in C++
3. Write a program that shows defining member function inside and outside of class body and demonstrate use of inline function
4. Write a program to implement function overloading concept
5. Write a program to implement all types of constructors and destructor also demonstrate constructor overloading
6. Write a program that shows use of static data member and static member function.
7. Write a program that shows use of nesting classes.
8. Write a program that shows passing and returning object from function.
9. Write a program that shows explicit type conversion
10. Write a program to overload different unary and binary operators by using friend and member function.
11. Generate the result for 5 students with following data - Name, exam no, Theory marks in5 subjects, grade(Use array of object concept).
12. Write a program to demonstrate friend function, friend class, member function of a classis friend to another class.
13. Write a program to count no. of objects created by using static data member & memberfunction.
14. Write a program to overload unary operators (++, - -, -).
15. Write a program to overload binary operator.(+, -, *, /, %) by using member function andfriend function.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level							
	Sem:	I					
	Paper Category:	DSC2-1 **(Major)**					
	Paper Name:	Python Programming-I					
	Credit:	02			**Theory:**	2 Hrs./Week	
Marks:	**UA:**	30	**CA:**	20	**Total:**	50	

Course Objective:

1. To learn the fundamentals of Python programming
2. To learn different data structures used in Python
3. To learn different control statements used in logic development.
4. To learn the various operations on the array, list, tuple, string, set, and dictionary.

Course Outcomes:

Upon successful completion of this course, students will be able to

1. Understand the basic concepts and applications of Python.
2. Design, create, build, and debug Python applications.
3. Explore the Integrated Development Environment (IDE).
4. Write and apply decision structures for different operations.
5. Write loop structures to perform iterative tasks.

Unit-I: Basic of Python **[12]**

Introduction: Features of Python, Python Virtual Machine, Memory management, Garbage Collection, Installation of Python, setting the path to operating system environment, writing the first Python program, executing a Python program.

Datatypes in Python: Datatypes-Numeric, Sequence Type-String, List, Tuple, Boolean, Set, Dictionary, Binary Types, Type conversion- implicit and explicit, Python comments, literals, constants, Identifiers, naming conventions, operators, operator precedence and associativity, input and output statements, command-line arguments.

Control Statements: if statement, if..else statement, if..elif..else statement, while loop, for loop, else suite, infinite loop, nested loops, word indentation, break statement, continue statement, pass statement, assert statement, return statement.

Unit-II: Data Collection and Function **[18]**

String, List, Tuple, Set and Dictionary: Creating string, manipulating different operations on string, creating list, manipulating different operations on list, list comprehensions, creatingtuple, manipulating different operations on tuple, creating set, manipulating different operations on set, creating dictionary, manipulating different operations on dictionary.

Functions: Difference between function and method, defining a function, calling function, returning result from a function, returning multiple values from a function, functions are objects, formal and actual arguments, types of arguments, local, nonlocal and global variables, global keyword, recursive functions, anonymous functions or lambdas, using lambdas with filter(), map() and reduce() functions

Arrays in Python: Introduction, advantages of array, creating an array, types of arrays, importing array module, indexing and slicing on arrays, methods of array module.

Reference Books:

1. Python: The Complete Reference by Martin C. Brown.
2. Core Python Programming, Dreamtech publications, by R. Nageswara Rao.
3. Python Programming, A modular approach, First Edition, Pearson, by Taneja Sheetal
4. Learning with Python, Dreamtech publications, by Allen Downey
5. Python Programming for the Absolute Beginner by Michael Dawson-Cengage Learning.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
	Sem:	I			
	Paper Category:	Practical **(Major)**			
	Paper Name:	Practical based on DSC2-1			
	Credit:	02		**Practical:**	4 Hrs./Week
Marks:	**UA:** 30		**CA:** 20		**Total:** 50

1. Write a simple Python script that prints "Hello, World!" to the console.
2. Perform basic arithmetic operations (addition, subtraction, multiplication, division) on numeric variables.
3. Concatenate strings and convert between data types as needed.
4. Implement conditional statements (if-else, elif) to control program flow based on different conditions.
5. Use loops (for, while) to iterate over lists, tuples, dictionaries, and ranges.
6. Create nested loops and conditional statements for more complex control flow logic.
7. Create and manipulate lists, tuples, dictionaries, and sets.
8. Access elements in data structures using indexing and slicing operations.
9. Use list comprehensions and generator expressions for concise data manipulation.
10. Write a Python program to find the sum of a list of numbers using a for loop.
11. Write a Python program to display stars in right angled triangular form using nestedfor loops.
12. Write a Python program to display a multiplication table from 1 to 10 using nested forloops.
13. Write a Python program to display elements in a list in reverse order.
14. Write a Python program to accept elements in the form of a tuple and display theirsum and average.
15. Write a Python program to create a dictionary with employee details and retrieve thevalues upon giving keys.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
Sem:	I				
Paper Category:	DSC3-1 **(Major)**				
Paper Name:	Fundamental of Computers				
Credit:	02	**Theory:**	2 Hrs./Week		
Marks:	**UA:** 30	**CA:** 20	**Total:** 50		

Course Objectives:

1. The main objective is to introduce IT in a simple language to all undergraduate students, regardless of their specialization.
2. It will help them to pursue specialized programs leading to technical and professional careers and certifications in the IT industry.
3. The focus of the subject is on introducing skills relating to IT basics, computer applications, etc.

Course Outcomes:

At the end of this course, the student should be able to

1. To understand basic concepts and terminology of information technology.
2. To a basic understanding of personal computers and their operations.
3. To understand various input and output devices.
4. To understand memory management.

Unit I: Introduction to Computers [15]

Basics of Computers:-Definition and characteristics of computers, History and evolution of computers, Generations of computers, Components of a Computer System:-Hardware and software, Input and output devices, Storage devices, CPU: ALU, CU, and registers, Types of Computers:-Analog, digital, and hybrid computers, Classification by size: micro, mini, mainframe, and supercomputers, Computer Software:-System software: Operating systems, utilities, Application software: Types and examples, Software development and programming languages, Operating Systems:-Types of operating systems, Examples of operating systems: Windows, Linux, MacOS, Android, Function of OS, Computer Networks:-Types of networks: LAN, WAN, MAN, PAN, Internet and its services, Network topologies, Data Representation:-Number systems: Binary, decimal, octal, hexadecimal, types of logic gates (AND, OR, NOT, NAND, NOR, XOR, XNOR), DeMorgan's Theorem, 1s & 2s complement, Binary arithmetic: Addition, subtraction, multiplication, division, Data representation: ASCII, Unicode,

Unit II: Computer Hardware and Software [15]

Input Devices:- Keyboard, mouse, scanner, joystick, microphone, webcam, Specialized input devices: barcode reader, biometric sensor, Output Devices:-Monitor, printer, speakers, projector, Types of monitors: CRT, LCD, LED, Types of printers: Dot matrix, inkjet, laser, Storage Devices:-Primary storage: RAM, ROM, cache, Secondary storage: Hard disk, SSD, optical disks (CD/DVD/Blu-ray), USB drives, Central Processing Unit (CPU):-CPU architecture, Types of processors: Single-core, multi-core, GPU, Performance factors: Clock speed, cache size, instruction set, Algorithm, Flowchart, Pseudocode Introduction to Programming:- Definition and types of programming languages, Programming paradigms: Procedural, object-oriented, functional,

Reference Books: -

1. Computer Fundaments - P.K. Sinha, BPB Publications, Edition-4th, 2004.
2. Fundamental of computers - V. Raja Raman, PHI Learning; 6th edition, 2014.
3. Computer Today – Donaid N. Sanders.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
Sem:	I				
Paper Category:	Practical **(Major)**				
Paper Name:	Practical based on DSC3-1				
Credit:	02	**Practical:**	4 Hrs./Week		
Marks:	**UA:** 30	**CA:** 20	**Total:** 50		

1. Connect and use different input devices (keyboard, mouse, scanner, joystick, microphone, webcam) and output devices (monitor, printer, speakers, projector). Document their usage and functionality.
2. Install and explore the features of different operating systems (Windows, Linux, MacOS, Android). Write a comparative analysis highlighting their functions, advantages, and use cases.
3. Set up a small LAN and demonstrate file sharing. Draw diagrams of various network topologies (star, bus, ring, mesh) and describe their advantages and disadvantages.
4. Write a program to convert numbers between binary, decimal, octal, and hexadecimal systems.
5. Write a program to implement binary addition, subtraction, multiplication, and division
6. Write a program to implement basic logic gates (AND, OR, NOT, NAND, NOR, XOR, XNOR) using Python functions.
7. Write a program to implement DeMorgan's Theorem using the logic gate functions.
8. Write a program to perform 1s and 2s complement operations for binary numbers.
9. Write a program to convert text to its ASCII and Unicode values and vice versa.
10. Write simple programs in a chosen programming language (e.g., Python) demonstrating procedural, object-oriented, and functional programming paradigms.
11. Install and format a hard disk, SSD, and optical disk. Perform data transfer operations and compare their read/write speeds.
12. Write an algorithm for a simple problem, draw its flowchart, and convert it into pseudocode. Implement the algorithm in a programming language of your choice.
13. Write a script to capture and log keystrokes and mouse movements.
14. Connect and configure speakers and projectors. Play audio and video files, documenting the quality and settings used.
15. Burn data onto a CD/DVD/Blu-ray disc and verify the integrity of the data.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level						
Sem:	I					
Paper Category:	DSC4-1 **(Major)**					
Paper Name:	Software Engineering					
Credit:	02	**Theory:**	2 Hrs./Week			
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

The aim of this course is to prepare learners for a foundational understanding of computers, encompassing:

1. To understand the fundamental concepts and characteristics of systems.
2. To categorize different types of systems and understand system analysis.
3. To explore various SDLC models and their applications with different functional and non-functional requirements.
4. To learn various techniques for gathering system requirements.
5. To understand the process of designing and implementing a system.
6. To understand coding standards, size measures, complexity analysis, and verification.
7. To grasp the fundamentals of software testing.
8. To learn about software implementation and the maintenance process.

Course Outcomes-

1. Students will be able to define a system, identify its characteristics, and describe various types of systems.
2. Students will be able to describe and compare models such as the Waterfall model, V-shape model, Spiral model, Prototyping, Incremental, RAD, and Agile methodologies.
3. Students will be able to identify, document, and analyze user and system requirements.
4. Students will be proficient in conducting interviews, questionnaires, record reviews, and observations for requirement gathering.
5. Students will be able to design data flow diagrams, entity-relationship diagrams, structured charts, and create a data dictionary. They will also learn input and output design.
6. Students will learn to write clean, maintainable code, estimate effort and cost, and verify code correctness.
7. Students will be able to perform different types of testing and understand testing methodologies.
8. Students will understand the steps involved in implementing software and the different types of software maintenance.

Unit I: - **Introduction to Software Engineering** **[15]**

System concepts: Introduction system, characteristics, Elements of system, Types of system, System Analysis, Role of System Analyst, Software Engineering: Definition, Characteristics of software, Qualities of software. System Development life cycle: Waterfall model, V-shape model, Spiral model, Prototyping, incremental, RAD, Agile. Software requirements: Functional, Non-functional requirements, User requirement, System requirements, Fact finding techniques: Interviews, Questionnaire, Record reviews, Observation Analysis and Design Tools: Flow charting, Decision tables, Decision Trees, Structured English, Structure charting Techniques.

Unit II: - **System Design and Implementation, Maintenance** **[15]**

Data flow Diagram (Physical, Logical), Entity relation diagram, structured chart, Data Dictionary, Input and output design, Types of Dependencies, Normalization (1NF,2NF,3NF,BCNF,4NF,5NF) Coding: Verification, size measures, complexity analysis, coding standards, Effort Estimation, Cost Estimation, Testing fundamentals Construction of the system: traditional and incremental approaches, conversion methods, Software Implementation, Overview of maintenance process, types of maintenance.

Reference Books-

1. Analysis and Design of Information Systems By James Senn.
2. Practical guide to structure System Design By Miller/Page/jones.
3. Software Engineering By Pressman.
4. System Analysis and Design By Parthsarty

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level						
Sem:	I					
Paper Category:	Practical **(Major)**					
Paper Name:	Practical based on DSC4-1					
Credit:	02		**Practical:**	4 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

1. Perform a system analysis for a given case study. Identify and document the system's components, processes, and interactions.
2. Create a checklist or template to evaluate the qualities of software (correctness, reliability, efficiency, usability, maintainability, portability) for a given application.
3. Create flowcharts or diagrams for different SDLC models (Waterfall, V-shape, Spiral, Prototyping, Incremental, RAD, Agile). Compare their advantages and disadvantages in a report.
4. Develop a requirements specification document for a small software project, including functional, non-functional, user, and system requirements.
5. Document requirements for an online shopping system.
6. Design a questionnaire and conduct interviews or observations to gather requirements for a proposed system. Summarize your findings in a report.
7. Create a flowchart for a simple process using any tool. Document the steps involved in creating the flowchart.
8. Develop a decision table and decision tree for a given set of business rules.
9. Develop a structure chart for a software module using a tool. Explain the hierarchy and interactions between modules.
10. Create a DFD and ERD for a Library Management System using a tool. Include entities, attributes, and relationships and create a data dictionary for a small database system. Include definitions, data types, and relationships for all data elements. Also design input forms and output reports for a given system.
11. Normalize a set of database tables to 1NF, 2NF, 3NF, BCNF, 4NF, and 5NF.
12. Use techniques like COCOMO or Function Point Analysis to estimate the effort and cost for a small software project. Document your estimation process and results.
13. Develop test cases for a given software module. Perform unit testing, integration testing, and system testing using tools like JUnit or Selenium.
14. Design a conversion plan for migrating data from an old system to a new system. Discuss parallel, direct, phased, and pilot conversion methods.
15. Develop a deployment plan for a software application. Include steps for installation, configuration, and user training and write an overview of the software maintenance process. Describe different types of maintenance (corrective, adaptive, perfective, preventive) and provide examples for each type.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
Sem:	I				
Paper Category:	DSC5-1 **(Major)**				
Paper Name:	Digital System				
Credit:	02	**Theory:**	2 Hrs./Week		
Marks:	**UA:** 30	**CA:** 20	**Total:** 50		

Course Objectives:

1. To provide a comprehensive introduction to the fundamental axioms, theories andconventions underlying the operation of digital systems.
2. To equip students with the necessary skills which will allow them to analyse, design,test, and simulate the operation of basic digital circuits.
3. To utilize a variety of digital logic design and simulation tools.

Course Outcomes:

At the end of this course, the student should be able to

1. Identify a digital system and its main characteristics, and differentiate between digitaland analogue systems
2. Describe the concepts of binary numbers and binary encoding, and performconversions between binary, decimal, and hex numbers and between binary codes
3. Perform basic mathematical operations using binary numbers, and design digitalsystems capable of performing such operations.
4. Describe theorems and axioms of Boolean Algebra, and utilize them effectively in theprocess of designing digital systems.
5. Model, analyse, design, test, and simulate the operation of combinational andsequential circuits using analytic and modular methodologies and tools.
6. Explain the concept of memory in digital systems, and design basic memory modules.

Unit I: Basic of Digital System **[15]**

History and Overview: History, applications of digital systems, digital signals and analog signals, advantages and disadvantages of digital systems.

Number System: The binary, decimal and hexadecimal number system, conversion of binary,decimal and hexadecimal number system, Signed and unsigned binary numbers, representing signed binary numbers using the 2's complement method, performing basic mathematical operations using binary numbers (addition, subtraction, multiplication, division), design of adder and subtractor circuits.

Block diagram – ALU, memory unit, control unit, motherboard, SMPS, expansion slots, serial and parallel ports.

Unit II: Boolean Algebra & Microprocessor **[15]**

Boolean Algebra fundamentals: definition of boolean algebra and logic gates, types of logicgates (AND, OR, NOT, NAND, NOR, XOR, XNOR), DeMorgan's Theorem, implementationof logic circuits using boolean equation and truth table.

Microprocessor: History and overview, comparative study of 8085, 8086 and Pentiumprocessor, architecture of Pentium microprocessor, features of Pentium microprocessor, applications of Pentium, addressing modes, instruction set, types of programming languages, assembly language programming, applications of Pentium.

Reference Books:

1. Digital Systems: Principles and Applications by Ronald J. Tocci, Neal S. Widmer, andGreg Moss
2. Digital Design: With an Introduction to the Verilog HDL, VHDL, and SystemVerilogby M. Morris R. Mano, and Michael D. Ciletti
3. Fundamentals of Digital Logic with VHDL Design, by Stephen Brown and ZvonkoVranesic
4. Digital Fundamentals, by Thomas L. Floyd
5. Digital Logic Design, by B. Holdsworth
6. System Programming byJohn J. Donowon
7. System programming and Operating System by Dhamdhere
8. System Software by Beck

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
Sem:	I				
Paper Category:	Practical **(Major)**				
Paper Name:	Practical based on DSC5-1				
Credit:	02	**Practical:**	4 Hrs./Week		
Marks:	**UA:** 30	**CA:** 20	**Total:** 50		

Tools / Software: NASM (Netwide Assembler)/ MASM (Microsoft MacroAssembler) / TASM (Turbo Assembler)etc.

1. Write an ALP to add 2 Multibyte no.
2. Write an ALP to subtract two Multibyte numbers.
3. Write an ALP to multiply two 16-bit numbers.
4. Write an ALP to divide two numbers.
5. Write an ALP to multiply two ASCII no.
6. Develop and execute and assembly language program to perform the conversion fromBCD to binary.
7. Write an ALP to convert binary to BCD.
8. Write an ALP to find the square of a number.
9. Write an ALP to find the cube of a number.
10. Write an ALP to separate odd and even numbers
11. Write an ALP to separate positive and negative numbers
12. Write an ALP to find logical ones and zeros in a given data
13. Write an ALP to demonstrate the AND Gate, OR Gate,
14. Write an ALP to demonstrate the NOT Gate, NAND Gate
15. Write an ALP to demonstrate the NOR Gate, XOR Gate.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level						
Sem:	I					
Paper Category:	SEC1					
Paper Name:	Introduction to Web Design					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objective:

1. Give the distinguishing characteristics of scripting language.
2. Discuss the reasons for and effects of nonstandard client-side scripting languagecharacteristics, such as limited data types, dynamic variable types and properties, and extensive use of automatic type conversion.
3. Develop event-driven programs that use HTML intrinsic event attributes, DOM events, listeners, and DOM-generated events.
4. Use the DOM to modify a document's attributes and style properties as well as to modify its parse-tree representation.

Course Outcomes:

1. Explain the history of the internet and related internet concepts that are vital inunderstanding web development.
2. Discuss the insights of internet programming and implement complete applicationsover the web.
3. Demonstrate the important HTML tags for designing static pages and separate designfrom content using Cascading Style sheet.
4. Utilize the concepts of JavaScript.

Unit I: Basic Web Design **[10]**

Introduction to Web Design: Introduction to Networking, Introduction to Internet, Applications of the Internet. Introduction to HTML, Structure of HTML, Creating and opening HTML file, Singular and paired tags, Text formatting tag, Anchor tag, Lists, Image, Image Map, Table, Frames and Frameset, form.

Unit II: CSS and JavaScript **[20]**

Introduction to CSS and JavaScript: Introduction to CSS, Types of CSS, Use of CSS, Selectors, Properties, Values.

CSS Properties: Background, Text, Fonts, Link, List, Table, Box Model, Border, Margin, Padding, Display, Positioning, Floating, Opacity, Media type, Backgrounds, Animations, Multiple Column Layout, Navigation bar.

Introduction to JavaScript- JavaScript Variables and Data types, Operators, Built-in functions in JavaScript, Control structure in JavaScript, JavaScript Objects- Object, Array, String, Date, Math, Number, Boolean Introduction to Java Script DOM and BOM, difference between DOM and BOM, DOM Objects- Document Object, Element Objects, Node Objects, BOM Objects-Window, Navigator, Location, Document, user-defined function, Form Validation, event and event handling in JavaScript.

Reference Books:-

1. HTML5 Black Book Kogent Learning Solutions Inc Dream-tech.
2. Beginning JavaScript and CSS Development with jQuery Richard York.
3. Beginning HTML and CSS Rob Larsen

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
Sem:	II				
Paper Category:	DSC1-2 **(Major)**				
Paper Name:	OOP'S with C++-II				
Credit:	02	**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:** 50

Course Objective: Students will try to learn-

1. Inheritance and Polymorphism in C++
2. Files management and templates in C++
3. Handling exceptions to control errors.

Course Outcomes: Upon successful completion of this course, students will be able to

1. Explain Inheritance, Polymorphism and create programs using them
2. Describe and use constructors and destructors
3. Understand and employ file management
4. Demonstrate how to control errors with exception handling.

Unit I: Inheritance and Runtime Polymorphism [12]

Introduction to inheritance, benefits, defining derived class, Types of derivations-Public, Private and Protected, Types (Forms) of Inheritance- Single, Multi-level, Multiple, Hierarchical, Hybrid, Multi- path (Virtual base class), Behavior of constructors and destructor in inheritance, Overloaded member functions, Pointer to base class, Pointer to derived class, Object composition-delegation

Runtime polymorphism: Introduction to runtime polymorphism, Virtual functions- Characteristics, Use of virtual function, Pure virtual function-Characteristics, Use, Abstract class, virtual destructors

Unit II: Stream and Files: [18]

Introduction to streams in C++, Stream classes, File stream classes, Formatted and unformatted I/O functions and Manipulators.

File Manipulations- Opening, closing, reading, writing, Appending, File opening modes-Opening files, using open() and constructor, Error handling during file manipulations, Command line arguments.

Exception Handling and Template:

Introduction to Exception handling, Exception handling mechanism-try, catch, throw keywords, Custom exception. Introduction to Function and Class template, inheritance of class template, class template containership.

Reference Books:

1. OOP in C++ – E-balagurusamy
2. Mastering C++ - K.R. Venugopal
3. Structured approach using C++ – Behrouz A. Forouzan
4. The Complete Reference C++- Fourth Edition. Herbert Schildt

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
Sem:	II				
Paper Category:	Practical **(Major)**				
Paper Name:	Practical based on DSC1-2				
Credit:	02		**Practical:**	4 Hrs./Week	
Marks:	**UA:** 30		**CA:** 20	**Total:** 50	

1. Write a program to implement all types of inheritance.
2. Write a program that shows use of pointer to base class and derived class
3. Write a program that shows use of virtual function and pure virtual function.
4. Write a program that shows use of abstract class
5. Write a program that shows use of virtual destructor
6. Write a program that shows behavior of constructor and destructor in inheritance.
7. Write a program that shows use of istream and ostream class.
8. Write a program that shows use of different manipulators.
9. Write a program to read, write and append data into file.
10. Write a program that checks two files are identical or not.
11. Write a program that shows use of random access of file.
12. Write a program that shows use of command line argument.
13. Write a program that shows use multiple catch blocks.
14. Write a program that shows use of custom exception.
15. Write a program that shows use of function template and class template

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level						
Sem:	II					
Paper Category:	DSC2-2 **(Major)**					
Paper Name:	Python Programming -II					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objective:

1. To learn the use of functions in programming.
2. To understand the use of modules and packages in the application hierarchy.
3. To understand Python programming using object-oriented programming principles.
4. To learn handling of various exceptions during the application development.
5. To understand the working with different file operations.

Course Outcomes: Upon successful completion of this course, students will be able to

1. Write and implement a functional approach to application development.
2. Write and implement a modular approach to application development.
3. Design an application using an object-oriented paradigm.
4. Create error-free applications by applying the exception-handling concept.
5. Design an application that contains the use of different files for data processing.

Unit I: Modules and packages **[15]**

Introduction to modules and packages, import statement, from…import statement, creating our own modules, working with built-in modules- Math module, time module and random module.

Python Object Oriented: Difference between procedure-oriented and object-oriented programming. Features of object-oriented programming- classes and objects, inheritance, polymorphism, encapsulation, abstraction. Creating class, self-variable, constructor, types of variables, namespaces, types of methods, passing member of one class to another class, inner classes. Types of inheritance, super() method, method overloading, method overriding, operator overloading, abstract classes, and interfaces.

Unit II: Exception Handling and Threading **[15]**

Threading: Understanding threads, Class and threads, Creating Threads, Thread Synchronization, Treads Life cycle, Multi-threading.

Exception Handling: Error in Python program, exceptions, steps in exception handling using try, except, else and finally blocks, types of exceptions- built-in and user-defined exceptions, assert statement.

File Input Output: Types of files in Python, opening a file- the file opening modes, closing a file, working with text files containing strings, working with binary files, with statement, pickling and unpickling, seek() and tell() methods, random accessing of binary files, zipping and unzipping files, working with directories.

Reference Books:

1. Python: The Complete Reference by Martin C. Brown.
2. Core Python Programming, Dreamtech publications, by R. Nageswara Rao.
3. Python Programming, A modular approach, First Edition, Pearson, by Taneja Sheetal
4. Learning with Python, Dreamtech publications, by Allen Downey
5. Python Programming for the Absolute Beginner by Michael Dawson-Cengage Learning.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
Sem:	II				
Paper Category:	Practical **(Major)**				
Paper Name:	Practical based on DSC2-2				
Credit:	02	**Practical:**	4 Hrs./Week		
Marks:	**UA:** 30	**CA:** 20	**Total:** 50		

1. Write a Python program to create a module and import it.
2. Write a Python program to create a package and import it.
3. Write a Python program to demonstrate the instance method, class method and static method.
4. Write a Python program to demonstrate inner classes.
5. Write a Python program to demonstrate Constructors in Inheritance.
6. Write a Python program to demonstrate method overriding.
7. Write a Python program to demonstrate unary and binary operator overloading.
8. Write a Python program to read, write and append the data from the text file and display them.
9. Write a Python program to count a number of lines, words and characters in a file.
10. Write a Python program to demonstrate all inbuild exception.
11. Write a Python program to demonstrate user defined exception.
12. Write a python program to illustrate the use of raising an exception
13. Write a program for Thread Synchronization
14. Write a program to demonstrate multi-threading.
15. Write a program to demonstrate thread life cycle.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level					
Sem:	II				
Paper Category:	DSC3-2 **(Major)**				
Paper Name:	Basics of Operating System				
Credit:	02	**Theory:**	2 Hrs./Week		
Marks:	**UA:** 30	**CA:** 20	**Total:** 50		

Course Objective: Students will try to learn-

1. To understand the main components of an OS and their functions.
2. Describe the functions of a modern OS with respect to convenience, efficiency and the ability to evolve.
3. To make aware of different types of OS and their services.
4. To learn different process scheduling algorithms and synchronization techniques to achieve better performance of a computer system.

Course Objectives:

1. To provide a sound understanding of the Computer operating system, its structures, and its functioning.
2. To understand the services provided by and the design of an operating system.
3. To understand different approaches to memory management.
4. To understand the services provided by and the design of an operating system.
5. To understand what a process is and how processes are synchronized and scheduled.

Unit I: Basic of Operating system **[10]**

Definition of operating system, Types of Operating Systems-Batch, Multiprogramming, Time Sharing, Real-Time, Distributed, Parallel, OS Services, System components, System Calls.

Process Management: Introduction to Process, Process states, Process Control Block, Context switching, Operations on Process, Introduction to Thread, Difference between process and thread, types of thread.

Unit II: Scheduling and Process Synchronization **[20]**

Scheduling, Concept of Process Scheduling, Types of Schedulers, Scheduling criteria, Scheduling algorithms Preemptive and Non-preemptive, FCFS, SJF, Round Robin, Priority Scheduling, Multilevel Queue Scheduling, Multilevel- feedback Queue Scheduling.

Process Synchronization: The Producer Consumer Problem, Race Conditions, Critical Section Problem, Semaphores, Busy waiting, Spinlock, and Classical Problems of Synchronization: Reader-Writer Problem, Dinning Philosopher Problem.

Reference Books:

1. Operating System Concepts By Siberchatz and Galvin.
2. Modern O.S. By Andrews Tanenbaum.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level						
Sem:	II					
Paper Category:	Practical **(Major)**					
Paper Name:	Practical based on DSC3-2					
Credit:	02		**Practical:**	4 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

1. Write a Python Program to implement the producer–consumer problem using semaphores.
2. Write a Python program to simulate the concept of the Dining-Philosophers problem.
3. Write a program to implement Threading and Synchronization Applications
4. Write a program for implementation of Priority scheduling algorithms
5. Write a program for implementation of Round Robin scheduling algorithms
6. Write a program for implementation of FCFS scheduling algorithms.
7. Write a program for implementation of SJF scheduling algorithms.
8. Write a program to simulate the concept of Dining-Philosophers problem.
9. Write a program to implement Threading and Synchronization Applications.
10. Write a program to implement banker's algorithm for deadlock avoidance.
11. Write a program to implement algorithm for deadlock detection.
12. Write a program for implementation memory allocation methods for fixed partition
13. Write a program to simulate the following contiguous memory allocation techniques
    a)Worst-fit b) Best-fit c) First-fit
14. Write a program to implement Paging technique for memory management.
15. Write a program for implementation of FIFO, LRU and LFU page replacement algorithm.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level						
Sem:	II					
Paper Category:	DSC4-2 **(Major)**					
Paper Name:	Linux and Shell Programming					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objective: -

Students will try to learn:

1. To introduce Basic Linux general purpose Commands
2. To learn different editor
3. To learn shell script concepts.
4. To learn file management and permission advance commands.
5. To learn awk, grap, perl scripts.

Course Outcomes: -

Students will be able to:

1. Identify the basic Linux general purpose commands.
2. Apply and change the ownership and file permissions using advance Linux commands.
3. Use the awk, grep, perl scripts.
4. Implement shell scripts.
5. Apply basic of administrative task.

Unit 1: - Introduction of Linux **(10)**

History of Linux, Architecture of Linux system & features, Kernel, Shell & its type, Difference between Windows and Linux. Linux Distributions, Working environments: KDE, GNOME, Xface4, Hardware requirement, Installation procedure of Linux, Create partitions, Configuration of X system Users & Groups Management:- Create Users, Create groups, Special groups, Assigning permissions to users and Groups, File and Directory permissions- chmod, chown, chgrp., Linux File System:-Hierarchy of File system, File System parts- Boot Block, Super Block, Inode, Block, Data Block, File types, Devices and Drives in Linux, Mounting devices (CD/DVD, usb, hard drive partition), file system

Unit 2: - Linux Command and Shell Programming (20)

I/O and Redirection, Piping, Linux commands File and directory Management Commands:-mkdir, rmdir, cd and pwd, file, ls, cat, more, less, File and Directory Operations: find, cp, mv, rm, ln etc, Printing the files - lpr, lpq, lprm etc., Filter Commands & Editor:- Filters: head, tail , pr, cut, paste, sort, uniq, tr, grep, egrep, fgrep, sed., Communication commands:- mesg, talk, write, wall, mail., Text Editors- vi, vim, Archive and File compression commands, Process Management: Shell process, Parent and children, Process status, System process, Multiple jobs in background and foreground, Changing process priority with nice. Listing processes, ps, kill, premature termination of process.,

Introduction to shell scripting, Writing and executing simple shell scripts, Variables, data types, and operators in shell scripting, Meta characters, Control structure, Loop structure and case statement, Writing and using shell functions, passing arguments to shell scripts and functions, Returning values from functions,

Reference Books :

1. Official Red Hat Linux Users guide by Redhat, Wiley Dreamtech India
2. Beginning Linux Programming by Neil Mathew & Richard Stones, Wiley Dreamtech India
3. Red Hat Linux Bible by Cristopher Negus, Wiley Dreamtech India
4. UNIX Shell Programming by Yeswant Kanethkar, BPB
5. Shell Scripting: Expert Recipes for Linux, Bash, and More" by Steve Parker
6. Classic Shell Scripting" by Arnold Robbins and Nelson H.F. Beebe
7. Learning the bash Shell: Unix Shell Programming" by Cameron Newham and Bill Rosenblatt

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level							
Sem:	II						
Paper Category:	Practical **(Major)**						
Paper Name:	Practical based on DSC4-2						
Credit:	02			**Practical:**	4 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50	

1. Navigate the file system using commands such as ls, cd, pwd, mkdir, and rmdir.
2. View file contents using commands such as cat, less, and head.
3. Manipulate files and directories using commands such as cp, mv, rm, and touch.
4. Use commands such as chmod and chown to change file permissions and ownership.
5. View running processes and their resource usage using commands such as ps, top
6. Manage processes using commands such as kill, killall, pkill, and pgrep.
7. Manage users and groups using commands such as useradd, usermod, groupadd, and passwd.
8. Write and execute a simple shell script using a text editor and the bash interpreter.
9. Use variables to store and manipulate data within a shell script.
10. Implement conditional statements in shell.
11. Pass arguments to functions and return values using the return statement.
12. Read input from files and process data using commands such as cat, grep, awk, and sed.
13. Use redirection operators (>, >>, <) to redirect input and output streams between files and commands.
14. Write Linux script for checking given number is prime, Armstrong and palindrome.
15. Write Linux script to display Fibonacci sequence up to n numbers.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level						
Sem:	II					
Paper Category:	DSC5-2 **(Major)**					
Paper Name:	Computational Mathematics					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

1. To introduce the concepts of mathematical logic.
2. To introduce the concepts of relations, counting and functions.
3. To perform the operations associated with counting, functions, and relations.
4. To use Graph Theory for solving problems.
5. To introduce the matrix and its operations

Course Outcomes-

1. Ability to apply mathematical logic to solve problems.
2. Understand sets, relations, counting, matrix, and graph.
3. Able to use logical notation to define and reason about fundamental mathematical concepts such as relations, counting, matrices and graphs.
4. Able to model and solve real-world problems using graphs and trees.

Unit I: - Logic and Proofs & Relations [15]

Logic and Proofs: Propositional logic, Applications of Propositional logic, propositional equivalences, Predicates and Quantifiers, Rules of inference.

Relations: Relations and their properties, Representing relation, Closures of relations, Partial orderings.

Unit II: - Counting, Graphs and Matrices: [15]

Counting: The basics of counting, the pigeonhole principle, Permutation and Combinations, Applications of recurrence relations, Solving recurrence relations, Divide and Conquer algorithms and recurrence relations.

Graphs: Graphs and Graphs models, Graph terminology and special types of graphs, Representing graphs and Graph isomorphism, Connectivity, shortest path problems & Dijkstra's algorithm.

Matrices: Introduction, operations, inverse, Rank of a matrix, solution of simultaneous linear equations, Eigen values and Eigen Vectors.

Reference Books:

1. Modern Algebra - S. Arumugam and A. Thangapandi Isaac, Scitech publications, 2005.
2. Invitation to Graph Theory- S.Arumugam and S.Ramachandran, Scitech Publications, 2005, Chennai.
3. Discrete Mathematical Structures with applications to Computer Science Tremblay and Manohar, McGrawHill, 1997.
4. Mathematical Structure for Computer Science, Discrete Mathematics and its Applications, Judith L.Gersting, W.H. Freeman and Company, Seventh Edition, 2014.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level							
Sem:	II						
Paper Category:	Practical **(Major)**						
Paper Name:	Practical based on DSC5-2						
Credit:	02		**Practical:**	4 Hrs./Week			
Marks:	**UA:**	30	**CA:**	20	**Total:**	50	

1. Write Python program for AND, OR, NOT, IMPLICATION, and BICONDITIONAL operation to Implement basic propositional logic operations.
2. Write a Python program to check the equivalence of two propositional expressions using truth tables.
3. Write a Python program to check if a relation is reflexive, symmetric, transitive, and antisymmetric and to verify if a relation is a partial order and find the Hasse diagram of a partial order.
4. Write a Python program to represent a relation as a matrix and perform basic operations on the matrix.
5. Write Python program to count permutations, combinations, and apply the principle of inclusion-exclusion and also generate permutations and combinations of a given set.
6. Write a Python program to solve problems using the pigeonhole principle (e.g., finding duplicates in a list).
7. Write a Python program to solve simple recurrence relations (e.g., Fibonacci sequence) and analyze their complexity.
8. Write Python functions to solve linear homogeneous recurrence relations with constant coefficients.
9. Write Python programs for merge sort and quicksort, and analyze their time complexity using recurrence relations.
10. Write a Python program to represent graphs using adjacency lists and adjacency matrices.
11. Write a Python program to check if two graphs are isomorphic using adjacency matrices.
12. Write Python programs for breadth-first search (BFS), depth-first search (DFS), and Dijkstra's algorithm to find the shortest path.
13. Write Python functions to perform matrix addition, subtraction, multiplication, finding the transpose and inverse and rank of a matrix.
14. Write a Python program to solve systems of linear equations using Gaussian elimination and LU decomposition.
15. Write a Python program to compute the eigenvalues and eigenvectors of a matrix using NumPy.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level							
Sem:	II						
Paper Category:	SEC2						
Paper Name:	**Advanced Web Designing**						
Credit:	02	**Theory:**	2 Hrs./Week				
Marks:	**UA:**	30	**CA:**	20	**Total:**	50	

Course Objective:

1. To identify the capabilities of JavaScript and jQuery and their role in web design and the document object model.
2. To respond to user events using jQuery, creating interactivity.
3. To provide a collection of syntax for template designs.
4. To develop and apply appropriate website or web application information architectures.
5. To design effective user interfaces.

Course Outcomes:

Upon successful completion of this course, students will be able to-

1. Understand the concepts of jQuery, bootstrap.
2. Build interactive web applications using JQuery and bootstrap.
3. Develop solution to complex problems using appropriate method, technologies, frameworks, web services and content management
4. Extend this knowledge to .Net Platforms, Java Technologies, Full Stack Development.

Unit I: JQUERY **[12]**

JQUERY: Introduction to jQuery, Difference between document Ready and Window Load, Adding jQuery to Your Web Pages, jQuery Syntax, jQuery Selectors, jQuery Event Methods, Effects-Hide and Show, Fading, Sliding, Animation, Button in jQuery, How to handle forms, Callback Functions, Chaining, Get and Set Content and Attributes, Add Elements, Add Several New Elements, Remove Elements, Get and Set CSS Classes, css() Method, The noConflict() Method

Unit II: BOOTSTRAP **[18]**

BOOTSTRAP: Introduction to Bootstrap Framework, History of Bootstrap, Advantages, Responsive web page, Major Features of Bootstrap, Setting up Environment, Syntax of Bootstrap, Introduction to Bootstrap container and its types, Introduction to Bootstrap Grid, Bootstrap Forms: Form Layouts, Inline Form, form controls (input, textarea, checkbox, radio, select), Static Control, Media Objects, Filters.

Bootstrap Components: Jumbotron, Button, Grid, Table, Alert , Wells , Badge and label, Panels , Input Groups, Input Types, Modals, Popover, Scrollspy, Pagination, Pager , Image , Glyphicon , Carousel , Progress Bar , List Group , Dropdown , Collapse, Tabs/Pill, Navbar,

Reference Books:

1. Nicholas C Zakas, "Professional JavaScript for Web Developers", 3rd Edition, Wrox/Wiley India, 2012.
2. David Sawyer Mcfarland, "JavaScript & jQuery: The Missing Manual", 1st Edition, O'Reilly/Shroff Publishers & Distributors Pvt Ltd, 2014.
3. Benjamin Jakobus and Jason Marah, Mastering Bootstrap 4, Packt Publishing, 2016.
4. Jacob Lett, Bootstrap 4 Quick Start: Responsive Web Design and Development Basics for Beginners, 2018.

B.Sc. (Entire Computer Science)-I, Level - 4.5 UG Certificate Level						
Sem:	II					
Paper Category:	GE1/OE1					
Paper Name:	Office Automation					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives:

1. To provide an in-depth training in use of office automation, internet and internet tools. The course also helps the candidates to get acquainted with IT.
2. To help the students to understand how to format, edit, and print text documents and prepare for desktop publishing.
3. To create various documents newsletters, brochures, making document using photographs, charts, presentation, documents, drawings and other graphic images.
4. To work with the worksheet and presentation software.

Course Outcomes:

1. At the end of this course, the student should be able to
2. Integrate both graphs and tables created in Microsoft Excel into a laboratory report in Microsoft Word.
3. Generate equations, sample calculations, and basic diagrams in Microsoft Word.
4. Input experimental data into Microsoft Excel.
5. Perform calculations in Microsoft Excel using both manually inputting formulas and built-in Functions.
6. Generate simple and effective tables and graphs to describe experimental data in Microsoft Excel.
7. Properly format and organize a formal laboratory report in Microsoft Word.

Unit-I: Introduction to Computer & MS-Word : [15]

Introduction to Computer: Applications of Computer, Advantages of Computer, Characteristics of Computer, Hardware and Software, Block diagram of computer

MS Word: Working with Documents -Opening and Saving files, Editing text documents, Inserting, Deleting, Cut, Copy, Paste, Undo, Redo, Find, Search, Replace, Formatting page and setting Margins, Converting files to different formats, Importing and Exporting documents, Sending files to others, Using Tool bars, Ruler, Using Icons, using help., **Formatting Documents:** Setting Font styles, Font selection- style, size, color etc, Type face - Bold, Italic, Underline, Case settings, Highlighting, Special symbols, Setting Paragraph style, Alignments, Indents, Line Space, Margins, Bullets and Numbering.. **Setting Page style:** Formatting Page, Page tab, Margins, Layout settings, Paper tray, Border and Shading, Columns, Header and footer, Setting Footnotes and end notes – Shortcut Keys; Inserting manual page break, Column break and line break, Creating sections and frames, Anchoring and Wrapping, Setting Document styles, Table of Contents, Index, Page Numbering, date and Time, Author etc., Creating Master Documents, Web page. Creating Tables: Table settings, Borders, Alignments, Insertion, deletion, Merging, Splitting, Sorting, and formula. , Drawing: Inserting Clip Arts, Pictures/Files etc., **Tools:** Word Completion, Spell Checks, Mail merge, Templates, Creating contents for books, Creating Letter/Faxes, Creating Web pages, Using Wizards, Tracking Changes, Security, Digital Signature. Printing Documents – Shortcut keys.

Unit-II: MS-Excel & MS Power point: [15]

MS Excel: Spread Sheet and its Applications, Opening Spreadsheet, Menus - main menu, Formula, Editing, Formatting, Toolbars, Using Icons, Using help, Shortcuts, Spreadsheet types. Working with Spreadsheets-opening, Saving files, setting Margins, Converting files to different formats (importing, exporting, sending files to others), Spread sheet addressing - Rows, Columns and Cells, Referring Cells and Selecting Cells – Shortcut Keys., **Entering and Deleting Data:** Entering data, Cut, Copy, Paste, Undo, Redo, Filling Continuous rows, columns, Highlighting values, Find, Search and replace, Inserting Data, Insert Cells, Column, rows and sheets, Symbols, Data from external files, Frames, Clipart, Pictures, Files etc, Inserting Functions, Manual breaks., **Setting Formula:** Finding total in a column or row, Mathematical operations (Addition, Subtraction, Multiplication, Division, Exponentiation), using other Formulae., **Formatting Spreadsheets:** Labeling columns and rows, Formatting- Cell, row, column and Sheet, Category- Alignment, Font, Border and Shading, Hiding/ Locking Cells, Anchoring objects, Formatting layout for Graphics, Clipart etc., Worksheet Row and Column Headers, Sheet Name, Row height and Column width, Visibility - Row, Column, Sheet, Security, Sheet Formatting and style, Sheet background, Colour etc, Borders and Shading ,Shortcut keys., **Working with sheets:** Sorting, Filtering, Validation, Consolidation, and Subtotal., **Creating Charts:** Drawing. Printing. Using Tools – Error checking, Formula Auditing, Creating and Using Templates, Pivot Tables, Tracking Changes, Security, Customization.

MS Power point: Presentation – Opening new presentation, Different presentation templates, setting backgrounds, selecting presentation layouts., **Creating a presentation:** Setting Presentation style, Adding text to the Presentation. Formatting a Presentation: Adding style, Colour, gradient fills, Arranging objects, Adding Header and Footer, Slide Background, Slide layout. Adding Graphics to the Presentation- Inserting pictures, movies, tables etc into presentation, Drawing Pictures using draw., **Adding Effects to the Presentation:** Setting Animation and transition effect. Printing Handouts, Generating Standalone Presentation viewer.

Reference Books:

1. Information Technology in Business: Principles, Practices, and Opportunities by James A Senn, Prentice Hall.
2. Technology and Procedures for Administrative Professionals by Patsy Fulton-Calkins, Thomson Learning.
3. Computer Fundamental MS Office-Including Internet and Web Technology: Anupama Jain, Avneet Mehra
4. The Complete Reference: Virginia Andersen, McGraw Hill
5. MS Office 2007 in a Nutshell: S. Saxena, Vikas Publications
6. MS-Office 2007 Training Guide: S. Jain, BPB Publications
7. Learning Computer Fundamentals, MS Office and Internet and Web Technology: D. Mai dasani. Reading, Vols. 1 and 2. Macmillan, 1975, Bhasker, W. W. S. and Prabhu, N. S.

Equivalent Subject for Old Syllabus B.Sc. (ECS) - I (Semester–I and II)

(NEP-2020) Semester-I

Sr. No.	Name of the Old Paper (w.e.f. 2022-2023)	Name of the New Paper (w.e.f. 2023-2024)
1.	Fundamental of Computer	Fundamental of Computer
2.	Basics of Operating System	Basics of Operating System (Sem-II)
3.	Programming using 'C'	No Equivalence
4.	Python – I	Python Programming – I
5.	Numerical Methods	No Equivalence
6.	Graph Theory	No Equivalence
7.	Basic Electronics	No Equivalence
8.	Advanced Electronics	No Equivalence

Semester-II

Sr. No.	Name of the Old Paper (w.e.f. 2022-2023)	Name of the New Paper (w.e.f. 2023-2024)
1.	Introduction to Web Technology	Introduction to Web Design
2.	Operating System	No Equivalence
3.	Object Oriented Programming using C++	OOP'S with C++-I (Sem-I)
4.	Python – II	Python Programming – II
5.	Linear Algebra	No Equivalence
6.	Discrete Mathematics	No Equivalence
7.	Digital Electronics and Microprocessor	No Equivalence
8.	Introduction to Microcontroller and Embedded System	No Equivalence