DEBRA THANA SAHID KSHUDIRAM SMRITI MAHAVIDYALAYA

Gangaram Chak, Chak Shyampur, Debra, West Bengal

PROPOSED SYLLABUS (DRAFT) OF

BACHELOR OF SCIENCE WITH COMPUTER SCIENCE (MULTIDISCIPLINARY STUDIES)

3-YEAR UNDERGRADUATE PROGRAMME

(w.e.f. Academic Year 2025-2026)

Based on

Curriculum & Credit Framework for Undergraduate Programmes (CCFUP), 2023 & NEP, 2020

Level	YR.	SEM	Course Type	Course Code	Course Title	Credit	L-T-P	Marks		
								CA	ESE	TOTAL
B.Sc. in Physical Sc./ Math. & Comp. Sc. with Computer Science	2nd		**SEMESTER-III**							
		III	Major (Disc.-A2)	UG/III/COMP/3/MJ-A2	**T: Data Structure; P: Practical** *(To be studied by the students taken Computer Science as Discipline-A)*	4	3-0-1	15	60	75
			Major (Disc.-A3)	UG/III/COMP/3/MJ-A3	**T: OOPS Using C++; P: Practical** *(To be studied by the students taken Computer Sc. as Discipline- A)*	4	3-0-1	15	60	75
			SEC 03	UG/III/COMP/3/SE-3P	**P: Web design with HTML & CSS** *(To be studied by the students taken Computer Science as Discipline-C)*	3	0-0-3	10	40	50
			AEC	UG/III/AEC-3T	Communicative English-1 (***common for all programmes***)	2	2-0-0	10	40	50
			MDC	UG/III/MDC/IT-3T	Multidisciplinary Course-3 (***to be chosen from the list***)	3	3-0-0	10	40	50
			Minor (Disc.-C1)	UG/III/COMP/3/MI-C3	**T: Digital Logic** *(To be studied by the students taken Computer Science as Discipline-C)*	4	3-0-1	15	60	75
					Semester-III Total	20				375
		IV	**SEMESTER-IV**							
			Major (Disc.-B2)	UG/IV/COMP/4/MJ-B2	**T: Data Structure; P: Practical** *(Same as like A2 for students taken Computer Science as Discipline-B)*	4	3-0-1	15	60	75
			Major (Disc.-B3)	UG/IV/COMP/4/MJ-B3	**T: OOPS Using C++; P: Practical** *(To be studied by the students taken Computer Sc. as Discipline-B)*	4	3-0-1	15	60	75
			Major (Elective) -1	UG/IV/COMP/4/MJE-01	**P: Problem Solving Using Python** *(To be studied by students taken Computer Sc. as Discipline-A)*	4	0-0-3	15	60	75
			AEC	UG/IV/AEC-2T	MIL-1 (***common for all programmes***)	2	2-0-0	10	40	50
			Minor (Disc.-C2)	UG/IV/COMP/4/MI-C4	**T: Data Structure; P: Practical** *(To be studied b the students taken Computer Sc. as Discipline- C)*	4	3-0-1	15	60	75
			Summer Intern.	IA	Internship / Apprenticeship- **activities to be decided by the Colleges following the guidelines to be given later**	4	0-0-4	-	-	50
					Semester-IV Total	22				400
					TOTAL of YEAR-2	42	-	-	-	775

MJ= Major Programme (Multidisciplinary), MN = Minor, A/B = Choice of Major Discipline; C= Choice of Minor Discipline; SEC = Skill

Enhancement Course, AEC = Ability Enhancement Course, MDC = Multidisciplinary Course, VAC = Value Added Course; CA= Continuous Assessment, ESE= End Semester Examination, T = Theory, P= Practical, L-T-P = Lecture-Tutorial-Practical, MIL = Modern Indian Language, ENVS= Environmental Studie

(Multidisciplinary Studies)

SEMESTER-III

UG/III/COMP/3/MJ-A2: Data Structure Credits 04

OBJECTIVE OF THE COURSE

- Gain knowledge of different types of data structures and their applications in computer science..
- Evaluate time and space complexity using growth rates, order notation, and space analysis techniques..
- Implement and manipulate single and multi-dimensional arrays and understand the concept of pointers and their applications.
- Develop stack-based applications, perform infix, postfix, and prefix conversions.
- Implement and analyze singly, doubly, and circular linked lists along with their various applications in memory management and self-organizing lists.
- Understand and implement different types of queues, including circular queues, de-queues, and priority queues.
- Implement different tree structures such as binary trees and AVL trees, along with tree traversal techniques.
- Apply linear and binary search techniques, implement various sorting algorithms.

UG/III/COMP/3/MJ-A2T: Data Structure Credits 03

Course Outline:

Module-I Introduction to Data Structures 04 Hrs.

Introduction to Algorithm and Flowcharts, Analysis for Time and Space Requirements, Order notation , Space Analysis of an Algorithm. Static and

Dynamic Memory Allocation.

Module- I Arrays and Pointers **05 Hrs.**

Introduction to Arrays, Single and Multi-dimensional Arrays, Pointer, Pointer to Structure, Sparse Matrices (Array and Linked Representation).

Module- II Stacks **08 Hrs.**

Implementing single / multiple stack/s in an Array; Prefix, Infix and Postfix expressions, Utility and conversion of these expressions from one to another; Applications of stack; Limitations of Array representation of stack.

Module- III Linked Lists **10 Hrs.**

Singly, Doubly and Circular Lists (Array and Linked representation); Normal and Circular representation of Stack in Lists; Self Organizing Lists; Skip Lists

Module- IV Queues **05 Hrs.**

Array and Linked representation of Queue, De-queue, Priority Queues

Module- VI Trees **15 Hrs.**

Introduction to Tree as a data structure; Binary Trees (Insertion, Deletion , Recursive and Iterative Traversals on Binary Search Trees); Height-Balanced Trees (Various operations on AVL Trees). Tree traversal techniques.

Module- VII Searching, Sorting **08 Hrs.**

Linear Search, Binary Search, Comparison of Linear and Binary Search, Selection Sort, Insertion Sort, Bubble Sort.

UG/III/COMP/3/MJ-A2P: Data Structures Lab Credits 01

1. Implement a program to perform matrix addition, subtraction, and multiplication using arrays.
2. Implement different type of operations on array
3. Write a program to search an element from a list. Give user the option to perform Linear or Binary search.
4. Implement different type of operation on Singly Linked List.
5. Implement Singly, Doubly, Circular Linked List.
6. Perform Stack operations using Linked List and array implementation.

7. Implement queue operation using Linked List and array implementation.
8. Implement Linear Search and binary search.
9. Implement selection sort, bubble sort, insertion sort.

Suggested Readings:

1. Adam Drozdek, "Data Structures and algorithm in C++", Third Edition, Cengage Learning, 2012.
2. SartajSahni, Data Structures, "Algorithms and applications in C++", Second Edition, Universities Press, 2011.
3. Aaron M. Tenenbaum, Moshe J. Augenstein, Yedidyah Langsam, "Data Structures Using C and C++:, Second edition, PHI, 2009.
4. Robert L. Kruse, "Data Structures and Program Design in C++", Pearson, 1999.
5. D.S Malik, Data Structure using C++, Second edition, Cengage Learning, 2010.
6. Mark Allen Weiss, "Data Structures and Algorithms Analysis in Java", Pearson Education, 3rd edition, 2011
7. Aaron M. Tenenbaum, Moshe J. Augenstein, Yedidyah Langsam, "Data Structures Using Java, 2003.
8. Robert Lafore, "Data Structures and Algorithms in Java, 2/E", Pearson/ Macmillan Computer Pub, 2003
9. John Hubbard, "Data Structures with JAVA", McGraw Hill Education (India) Private Limited; 2 edition, 2009
10. Goodrich, M. and Tamassia, R. "Data Structures and Algorithms Analysis in Java", 4th Edition, Wiley, 2013
11. Herbert Schildt, "Java The Complete Reference (English) 9th Edition Paperback", Tata McGraw Hill, 2014.
12. D. S. Malik, P.S. Nair, "Data Structures Using Java", Course Technology, 2003.

UG/III/COMP/3/MJ-A3: OOPS Using C++ Credits 04

OBJECTIVE OF THE COURSE

- Understand the principles of Object-Oriented Programming (OOP) and compare it with structured programming.
- Develop proficiency in C++ fundamentals, including data types, control structures, functions, and user-defined data types.
- Implement encapsulation, constructors, destructors, and object manipulation techniques in C++.
- Apply concepts of polymorphism and inheritance, including operator overloading, virtual functions, and abstract classes.
- Utilize pointers, dynamic memory allocation, and exception handling for efficient program development.
- Gain hands-on experience in solving real-world problems using C++ through practical implementation of OOP concepts.

UG/III/COMP/3/MJ-A3T **Credits 03**

Module-I: Introduction to OOPs and C++ Element **15 Lectures.**

Structured vs. Object Oriented Programming, Object Oriented Programming Concepts, Benefits of Object oriented programming, Object Oriented Languages, Structure of a C++ program, Data Types, Operators and Control Structures, Iteration / Loop Construct, Arrays, Functions (User defined Function, Inline Function, Function Overloading), User Defined Data Types (Structure, Union and Enumeration).

Module II: Class, Object, Constructor & Destructor: 15 Lectures.

Defining Classes, Encapsulation, Instantiating Objects, Member Functions, Accessibility labels, Static Members, Friend Function, Purpose of Constructors, Default Constructor, Parameterized Constructors, Copy Constructor, Destructor.

Module III: Pointer, Polymorphism & Inheritance: 20 Lectures.

Pointer (Pointer to Object, this Pointer, Pointer to Derive Class), Introduction to Polymorphism (Compile time Polymorphism, Run time Polymorphism), Operator Overloading, Overloading Unary and Binary Operators, Virtual Function, Pure Virtual Functions, Inheritance (Single Inheritance, Multiple Inheritance, Multilevel Inheritance, Hierarchical Inheritance, Hybrid Inheritance), Virtual Base Class, Abstract Class.

UG/III/COMP/3/MJ-A3P: Credits 01

1. Write a C++ program to find the sum of individual digits of a positive integer.
2. Write a C++ program to print the given number in reverse order.
3. Write a C++ program to print first 100 non-Fibonacci numbers.
4. Write a C++ program to convert a decimal number into a hexadecimal number.
5. Write a C++ program to search an element of an array using binary search technique.
6. Write a C++ program to calculate compound interest in a bank using default arguments.
7. Write a C++ program to display the student details using classes and object as array.
8. Write a C++ program to implement stack using array.
9. Write a C++ program for matrix multiplication using dynamic memory allocation, copy construction and overloading of assignment operator.
10. Write a C++ program to read a two-dimensional matrix and display its transpose.
11. Write a C++ program to implement inline function.
12. Write a C++ program to implement constructor and destructor.
13. Write a C++ program to implement the functionalities of a copy constructor.
14. Write a C++ program to display the account number and balance using constructor overloading.
15. Write a C++ program to find the volume of cube, rectangle and cylinder using function overloading.
16. 16. Write a C++ program to overload operator ++ and operator - using friend functions.

Suggested Readings:

1. HerbtzSchildt, "C++: The Complete Reference", Fourth Edition, McGraw Hill.2003
2. BjarneStroustroup, "Programming -- Principles and Practice using C++", 2nd Edition, Addison- Wesley 2014.
3. E Balaguruswamy, "Object Oriented Programming with C++", Tata McGraw-Hill Education, 2008.
4. Paul Deitel, Harvey Deitel, "C++ How to Program", 8th Edition, Prentice Hall, 2011.
5. John R. Hubbard, "Programming with C++", Schaum's Series, 2nd Edition, 2000.
6. Andrew Koeni, Barbara, E. Moo, "Accelerated C++", Published by Addison-Wesley , 2000.
7. Scott Meyers, "Effective C++", 3rd Edition, Published by Addison-Wesley,2005.
8. Harry, H. Chaudhary, "Head First C++ Programming: The Definitive Beginner's Guide", First Create space Inc, O-D Publishing, LLC USA.2014
9. Walter Savitch, "Problem Solving with C++", Pearson Education, 2007.
10. Stanley B. Lippman, JoseeLajoie, Barbara E. Moo, "C++ Primer", Published by Addison-Wesley, 5th Edition, 2012
11. E Balagurusamy , Object Oriented Programming with C++, 5 th edition, Tata McGraw, 2011.
12. Deitel and Deitel, “C++: How to Program”, 9th Edition, Pearson, 2013.

SEC-3P (Skill Enhancement Course)

UG/III/COMP/3/SE-3P: Web design using HTML and CSS credits: 03

Course Objective:

- Develop a comprehensive understanding of key web technologies and the client-server architecture.
- Acquire proficiency in HTML and CSS, exploring diverse tags, elements, and the fundamental structure of HTML documents.
- Master the art of effective webpage styling using CSS, including selectors, properties, and layout techniques.
- Grasp the principles of responsive web design, mobile optimization, and media query implementation.
- Explore advanced features in HTML, including HTML5 elements, and delve into advanced CSS concepts and best practices.
- Create interactive forms, implement animations and transitions, and explore pseudo-classes and pseudo-elements using HTML and CSS.
- Recognize and implement web accessibility best practices, utilizing ARIA roles and attributes.
- Introduce and utilize CSS preprocessors.
- Deepen the ability to create advanced responsive layouts with CSS, exploring intricate layout techniques and media query

applications.

- Familiarize themselves with popular CSS frameworks like Bootstrap, integrating pre-built components and styles into web projects.

Course Outline:

1. **Introduction to Web Development**
    a. Overview of web technologies.
    b. Client-server architecture.
    c. Introduction to HTML and CSS.
2. **HTML Fundamentals**
    a. Program Explore HTML tags and elements.
    b. Understand the document structure in HTML.
    c. Learn about forms, multimedia, and semantic HTML.
3. **CSS Styling Techniques**
    a. Learn CSS for styling web pages.
    b. Explore CSS selectors and properties.
    c. Understand layout techniques.

4. **Advanced HTML and CSS**
    a. Explore advanced HTML features.
    b. Learn about HTML5 and its new elements.
    c. Dive into advanced CSS concepts and best practices.
5. **Interactive Elements with HTML and CSS**: **(5 Hours)**
    a. Create interactive forms using HTML.
    b. Implement animations and transitions with CSS.

HTML and CSS Practical:

- Create a simple webpage with headings, paragraphs, and a list.
- Design a form with various input types (text, password, radio buttons, and checkboxes).
- Build a table displaying information with proper headers and rows.
- Implement an ordered and unordered list to showcase a set of items.
- Develop a webpage with hyperlinks linking to different sections within the same page.
- Design a responsive navigation bar with dropdown menus.
- Create an HTML page that includes multimedia elements such as images, audio, and video.

- Develop a form that utilizes HTML5 semantic elements (e.g., <article>, <section>).
- Implement a webpage with an embedded Google Map.
- Style a webpage using internal CSS to change fonts, colors, and background.
- Create a CSS file and link it to an HTML file for external styling.
- Use CSS Grid to create a two-dimensional layout with rows and columns.
- Implement CSS transitions for smooth effects on hover or click events.
- Style a form with CSS to enhance its visual appeal.
- Customize the appearance of hyperlinks with different states (normal, hover, visited).

MINOR

UG/III/COMP/3/MI-C3: Digital Logic — Credit: 04

OBJECTIVE OF THE COURSE

- Learn the basics of binary, octal, decimal, and hexadecimal number systems and conversions between them.
- Master the principles of Boolean algebra, including logic operations, truth tables, and De Morgan's laws.
- Gain proficiency in simplifying Boolean expressions using techniques such as Karnaugh maps and the Quine-McCluskey method.
- Develop skills in designing basic combinational logic circuits, including adders, subtractors, multiplexers, and decoders.
- Understand the behaviour and design of sequential circuits, including flip-flops, latches, counters, and registers.
- Explore circuit minimization techniques to reduce complexity and cost in digital designs.

Number systems: **15 Hrs.**

Positional number systems; Binary, Octal, Hexadecimal, and Decimal number systems; conversion of a number in one system to the other; Representation of signed numbers-signed magnitude, one's complement, 2's complement representation techniques, Merits of 2's complement representation scheme; Various binary codes - BCD, excess -3, Gray code, binary addition and subtraction.

Boolean algebra: **15 Hrs.**

Fundamental of Boolean Expression: Definition of Boolean Algebra, Postulates, Basic Logic gates: (OR, AND, NOT); Universal Logic Gates: (NAND & NOR); Basic logic operations: logical sum (OR), logical product (AND), complementation (NOT), anti-coincidence (EX-OR) and coincidence (EX-NOR) operations: Truth tables of Basic gates; Boolean Variables and Expressions; De-Morgan's theorem; Boolean expressions Simplification- Algebraic technique, Karnaugh map technique, 3 variable and 4 variable Karnaugh map.

Combinational Circuits: **15 Hrs.**

Half Adder, Full Adder (3-bit), Half Subtractor, Full Subtractor (3-bit), and construction using Basic Logic Gates (OR, AND, NOT) and Universal Logic Gates (NAND & NOR), Multiplexer, Encoders, Demultiplexer, and Decoder circuits.

Sequential Circuits: **15 Hrs.**

Latch, RS, D, JK, T Flip Flops; Race condition, Master Slave JK Flip Flop; Registers: Serial Input Serial Output (SISO), Serial Input Parallel Output (SIPO), Parallel input Serial Output (PISO), Parallel Input Parallel Output (PIPO), Universal Shift Registers; Counters: Asynchronous Counter, Synchronous Counter.

Suggested Readings:

1. Morris Mano, Charles R. Kime, Logic and computer design fundamentals, Pearson Prentice Hall, 2004
2. Basavaraj,B., Digital fundamentals, New Delhi: Vikas Publishing House, 1999.
3. Kandel Langholz, Digital Logic Design, Prentice Hall, 1988.
4. Rafiquzzaman & Chandra, Modern Computer Architecture, West Pub. Comp., 1988.

SEMESTER-IV

UG/IV/COMP/4/MJ-B2: ***Same as like A2***
UG/IV/COMP/4/MJ-B3: ***Same as like A3***

UG/IV/COMP/4/MJE-01: Problem Solving Using Python **Credit: 04**

Course Objective:

- Introduce fundamental features of Python programming, focusing on industry standards.
- Enable students to apply advanced Python programming features to solve real-world problems.
- Instill a solid understanding of Python programming concepts, libraries.

Course Outline:

- Develop the capability to create applications in various fields using Python.
- Explore computer systems and the Python programming language.
- Emphasize computational thinking and cover Python data types, expressions, operators, variables, assignments, strings, lists, and the Python standard library.
- Dive into imperative programming, covering Python modules and built-in functions like print() and eval().
- Develop user-defined functions and assignments, emphasizing parameter passing.
- Focus on text data, files, and exceptions, covering strings, formatted output, files, errors, and exceptions.
- Introduce execution control structures, decision control, and the IF statement.
- Cover For LOOP and iteration patterns, including two-dimensional lists, while loop, additional loop patterns, and iteration control statements.
- String, List, Tuples, Dictionaries, Sets.

Python Practical:

- Write a menu driven program to convert the given temperature from Fahrenheit to Celsius and vice versa depending upon users' choice.
- Check if a number is prime
  or not.

- Write a Program to calculate total marks, percentage and grade of a student. Marks obtained in each of the three subjects are to be input by the user. Assign grades according to the following criteria:
  Grade A: Percentage >=80
  Grade B: Percentage>=70 and <80
  Grade C: Percentage>=60 and <70
  Grade D: Percentage>=40 and <60
  Grade E: Percentage<40

- Write a menu-driven program, using user-defined functions to find the area of rectangle, square, circle and triangle by accepting suitable input parameters from user.

- Write a Program to display the first n terms of Fibonacci series.
- Write a Program to find factorial of the given number.
- Write a Program to find sum of the following series for n terms: 1 – 2/2! + 3/3!--------- n/n!
- Write a Program to calculate the sum and product of two compatible matrices.
- Create two matrices and perform matrix multiplication using NumPy.
- Write a program to find the square root of a given number.

REFERENCE BOOKS:

1. "Python Programming: A Modular Approach with Graphics, Database, Mobile, and Web Applications" by Sheetal Taneja & Naveen Kumar, Pearson, 2017.
2. "Python: The Complete Reference" by Martin C. Brown, Osborne/McHraw Hill, 2001.
3. "Core Python Programming" by Wesley J. Chun, Pearson Education, Second Edition, 2007.
4. Introduction to Computing Using Python: An Application Development Focus" by Ljubomir Perkovic, John Wiley & Sons, 2012.

MINOR

UG/IV/COMP/4/MI-C4: ***Same as like A2***