# NEP

**Syllabus for**

# B.Sc. Computer Science

# Major,

# Minor

**and**

# Interdisciplinary

| Sem | | Major | Lab | Minor | Lab | Multidisciplinary | Lab | |
|---|---|---|---|---|---|---|---|---|
| I | 1 | Computer Fundamentals and programming with C | Y | Computer Fundamentals and programming with C | Y | Computer Fundamentals and programming with C | Y | |
| II | 2 | Data Structures using C++ | Y | Data Structures using C++ | Y | Data Structures using C++ | Y | |
| III | 3 | Basic Electronics and Digital Logic | Y | Operating Systems | Y | Operating Systems | Y | |
| IV | 4 | Object Oriented Programing | Y | | | Database Management System | Y | |
| | 5 | Operating Systems | Y | | | | | |
| | 6 | Computational Mathematics | N | | | | | |
| | 7 | Computer System Architecture | N | | | | | |
| V | 8 | Analysis of Algorithms | Y | | | Computer System Architecture | N | |
| | 9 | Database Management System | Y | | | | | |
| | 10 | Software Engineering | Y | | | | | |
| | 11 | Networking | N | | | | | |
| VI | 12 | Graphics and Multimedia | Y | | | Computer Networks | N | |
| | 13 | Artificial Intelligence | Y | | | | | |
| | 14 | Microprocessor and VLSI Design | Y | | | | | |
| | 15 | Theory of Computation | N | | | | | |
| VII | 16 | Image Processing and Computer Vision | Y | Database Management System | Y | | | |
| | 17 | Compiler Design | N | Computer System Architecture | N | | | |
| VIII | 18 | Machine Learning | Y | | | | | |
| | 19 | Cyber Security and IoT | N | | | | | |
| | 20 | Big Data | N | | | | | |
| | 21 | Minor Project | LAB | | | | | |
| | | | | | | | | |
| VIII | 20 | Big Data / Machine Learning / Cyber Security and IoT | N | | | | | Honors with Research |
| | 21 | Major Project | LAB | | | | | |
| | | | | | | | | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | MDC | Lab | SEC | Lab | | | |
| | | Computer Fundamentals | N | R Programming | Y | | | |
| | | Document Processing with Open Office | Y | Python Programming | Y | | | |
| | | Programming with Python | Y | Document processing with Latex | Y | | | |

## CORE COURSES

**CMSACOR01T: Computer Fundamentals and programming with C** **Theory: 45 Lectures**

**Overview of C** **(5 Lectures)**

History, Basic Structure, Algorithms, Structured programming constructs. Character sets, Tokens, Keywords, Constants, Variables, Data Types, Declaration of storage classes.

**Operators, Expressions and Preprocessor** **(8 Lectures)**

Arithmetic, Relational, Logical and Assignment; Increment and Decrement and Conditional, Bitwise, Special operator, Operator Precedence and Associativity; Arithmetic Expressions, Evaluation of expression, type casting. Comments, Input and output operations. Understanding the Preprocessor Directives (#include, #define, #error, #if, #else, #elif, #endif, #ifdef, #ifndef and #undef), Macros

**Decision and Loop Control Structure** **(7 Lectures)**

If-else statements, Nested if-else, switch, Conditional operator. While, do-While, for loop, break statements, continue statements, goto statements.

**Functions and Arrays** **(7 Lectures)**

Utility of functions, Call by Value, Call by Reference, Functions returning value, Void functions, Inline Functions, Return data type of functions, Functions parameters, Differentiating between Declaration and Definition of Functions, Command Line Arguments/Parameters in Functions, Functions with variable number of Arguments.

Creating and Using One Dimensional Arrays (Declaring and Defining an Array, Initializing an Array, Accessing individual elements in an Array, Manipulating array elements using loops), Use Various types of arrays (integer, float and character arrays / Strings) Two-dimensional Arrays (Declaring, Defining and Initializing Two Dimensional Array, Working with Rows and Columns), Introduction to Multi-dimensional arrays, return statement, return values and their types, String handling with arrays, String handling functions, recursion

**Pointers** **(6 Lectures)**

Definition and initialization, Pointer arithmetic, Pointers and arrays, String functions and manipulation, Dynamic storage allocation.

**User defined Datatypes and Memory Allocation** **(6 Lectures)**

Enumerated datatypes, Structures. Structure arrays, Pointers to Functions and Structures, Unions. Differentiating between static and dynamic memory allocation, use of malloc, calloc and free functions, use of new and delete operators, storage of variables in static and dynamic memory allocation

**File Access** **(6 Lectures)**

Opening and closing a file (use of fstream header file, ifstream, ofstream), Reading and writing Text Files,

Using put(), get(), read() and write() functions, Random access in files,

**Text Books**

1. Let Us C, Kanetkar, BPB Publication.
2. Programming in ANSI C, Balaguruswamy, McGraw Hill.
3. Programming with C, Byron S. Gottfried, McGraw Hill.

**Reference Books**

1. The C Programming Language, Kernighan and Dennis Ritchie, PHI.
2. The Complete reference C, Herbert Schildt, McGraw Hill.
3. Programming Languages, Allen B. Tucker, Tata McGraw Hill.

**CMSACOR01P: Computer Fundamentals and programming with C Lab** **Practical: 60 Lectures**

**Instruction: Use an open source C compiler.**

1. Write a program (WAP) to print the sum and product of digits of an integer.
2. WAP to reverse a non-negative integer.
3. WAP to compute the sum of the first n terms of the following series
   S = 1+1/2+1/3+1/4+……
4. WAP to compute the sum of the first n terms of the following series,
   S =1-2+3- 4+5…….
5. Write a function to find whether a given no. is prime or not. Use the same to generate the prime numbers less than 100.
6. Write a function that checks whether a given string is Palindrome or not. Use this function to find whether the string entered by the user is Palindrome or not.
7. WAP to compute the factors of a given number.
8. WAP to swap two numbers using macro.
9. WAP to print a triangle of stars as follows (take number of lines from user):

```
*
***
*****
*******
*********
```

10. WAP to perform following actions on an array entered by the user :
    a. Print the even-valued elements

b. Print the odd-valued elements
c. Calculate and print the sum and average of the elements of array
d. Print the maximum and minimum element of array
e. Remove the duplicates from the array
f. Print the array in reverse order

(The program should present a menu to the user and ask for one of the options. The menu should also include options to re-enter array and to quit the program.)

11. WAP that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments.
12. Write a program that swaps two numbers using pointers.
13. Write a program in which a function passes the address of two variables and then alter its contents.
14. Write a program which takes the radius of a circle as input from the user, passes it to another function that computes the area and the circumference of the circle and displays the value of area and circumference from the main() function.
15. Write a program to find the sum of n elements entered by the user. To write this program, allocate memory dynamically using malloc() / calloc() functions or new operators.
16. Write a menu driven program to perform following operations on strings:
    a. Show address of each character in string
    b. Concatenate two strings without using strcat function.
    c. Concatenate two strings using strcat function.
    d. Compare two strings
    e. Calculate length of the string (use pointers)
    f. Convert all lowercase characters to uppercase
    g. Convert all uppercase characters to lowercase
    h. Calculate number of vowels
    i. Reverse the string
17. Given two ordered arrays of integers, write a program to merge the two-arrays to get an ordered array.
18. WAP to display Fibonacci series (i) using recursion, (ii) using iteration.
19. WAP to calculate Factorial of a number (i) using recursion, (ii) using iteration.
20. WAP to calculate GCD of two numbers (i) with recursion (ii) without recursion.
21. Write a menu-driven program to perform following Matrix operations (2-D array implementation): a) Sum b) Difference c) Product d) Transpose
22. Copy the contents of one text file to another file, after removing all whitespaces.
23. Write a function that reverses the elements of an array in place. The function must accept only one pointer value and return void.
24. Write a program that will read 10 integers from the user and store them in an array. Implement an array using pointers. The program will print the array elements in ascending and descending order.
25. Add two distances in a meter kilometer system using structure.
26. Add two complex numbers using structures.
27. Calculate the difference between two time periods using structures.

**Introduction** **(5 Lectures)**

Data Object, Abstract Data Type, Data Structures and Data Types. Types of Data Structures – Linear and non-linear Data Structures.Single and Multi-dimensional Arrays, Address Calculations, Sparse Matrices (Array Representation).

**Linked Lists** **(7 Lectures)**

Singly, Doubly and Circular Lists (Array and Linked representation); Operations on Lists. Sparse Matrices (Linked Representation).

**Stacks and Queues** **(9 Lectures)**

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

**Recursion** **(5 lectures)**

Developing Recursive Definition of Simple Problems and their implementation; Advantages and Limitations of Recursion; Understanding what goes behind Recursion (Internal Stack Implementation)

**Binary Trees** **(10 Lectures)**

Introduction; Properties, Binary Trees Traversals (Recursive and Non-Recursive), Binary Search Trees (Insertion, Deletion), Recursive and Iterative Traversals on Binary Search Trees; Threaded Binary Trees (Concept only); Height-Balanced Trees (Concept only).

**Searching, Sorting and Hashing** **(9 Lectures)**

Linear Search, Binary Search, Comparison of Linear and Binary Search, Selection Sort, Insertion Sort, Bubble Sort, Comparison of Sorting Techniques. Introduction to Hashing, Resolving collusion by Open Addressing, Coalesced Hashing, Separate Chaining and simple examples.

**Text Books**


1. Aaron M. Tenenbaum, Moshe J. Augenstein, YedidyahLangsam, "Data Structures Using C and C++:, Second edition, PHI, 2009.
2. Sartaj Sahni, Data Structures, "Algorithms and applications in C++", Second Edition, Universities Press, 2011.


**Reference Books**


1. Robert L. Kruse, "Data Structures and Program Design in C++", Pearson,1999.
2. D. S. Malik, Data Structure using C++, Second edition, Cengage Learning, 2010.
3. Mark Allen Weiss, ″Data Structures and Algorithms Analysis in Java", Pearson Education, 3rd edition, 2011
4. Aaron M. Tenenbaum, Moshe J. Augenstein, Yedidyah Langsam, "Data Structures Using Java", 2003.
5. Samanta, D. "Classic data structures.", PHI, *Terminology* 2 (2001): 1.
6. Adam Drozdek, "Data Structures and algorithm in C++", Third Edition, Cengage Learning, 2012.


**CMSACOR02P:** Data Structures using C++ **Practical: 60 Lectures**

**Instruction: Use an open source C++ compiler.**

1. Write a program to search an element from a list. Give user the option to perform Linear

or Binary search.

2. WAP to sort a list of elements. Give user the option to perform sorting using Insertion sort, Bubble sort or Selection sort.
3. Implement Linked List. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists (include a function and also overload operator +).
4. Implement Doubly Linked List. Include functions for insertion, deletion and search of a number, reverse the list.
5. Implement Circular Linked List. Include functions for insertion, deletion and search of a number, reverse the list.
6. Perform Stack operations using Linked List implementation.
7. Perform Stack operations using Array implementation.
8. Perform Queues operations using Circular Array implementation.
9. Create and perform different operations on Double-ended Queues using Linked List implementation.
10. WAP to scan a polynomial using linked list and add two polynomials.
11. WAP to calculate factorial and to compute the factors of a given no. (i) using recursion, (ii) using iteration
12. WAP to display Fibonacci series (i) using recursion, (ii) using iteration
13. WAP to calculate GCD of two numbers (i) with recursion (ii) without recursion
14. WAP to create a Binary Search Tree and include following operations in tree:
    i. Insertion (Recursive and Iterative Implementation)
    ii. Deletion by copying
    iii. Deletion by Merging
    iv. Search a no. in BST
    v. Display its preorder, postorder and inorder traversals Recursively
    vi. Display its preorder, postorder and inorder traversals Iteratively
    vii. Display its level-by-level traversals
    viii. Count the non-leaf nodes and leaf nodes
    ix. Display height of tree
    x. Create a mirror image of tree
    xi. Check whether two BSTs are equal or not
15. WAP to convert the Sparse Matrix into non-zero form and vice-versa.
16. WAP to reverse the order of the elements in the stack using additional stack.
17. WAP to reverse the order of the elements in the stack using additional Queue.
18. WAP to implement Diagonal Matrix using one-dimensional array.
19. WAP to implement Lower Triangular Matrix using one-dimensional array.
20. WAP to implement Upper Triangular Matrix using one-dimensional array.
21. WAP to implement Symmetric Matrix using one-dimensional array.