Syllabi of Computer Science SEC Courses

Semester	**: I**
Course Type	**: SEC**
Course Code	**: CSCSEC101**
Name of the Course	**: Programming with C**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 30**
Total Marks	**: 100**
End Semester Marks	**: 80 [50 (Theory) + 30 (Lab)]**
Internal Marks	**: 20**

Course objectives:

1. *To introduce the concepts of programming and programming language C.*
2. *To explain the concepts of functions and programme structure in C*
3. *To explain how to write and implement C programs*
4. *To explain the concept and working of pointers and files in C*
5. *To introduce the low level programming in C*

UNIT-I

Fundamentals of computer programming with C – Data Types, Expressions, Operations – input, output; Writing simple C programs; Control structures (WHILE, DO-WHILE, FOR, IF-ELSE, SWITCH, BREAK, CONTINUE, GOTO STATEMENTS, nested loops etc.) and writing programs using control structures; solving elementary programming problems from various areas of applications including mathematics and statistics.

UNIT-II

Functions and program structure – Defining and accessing functions in C, passing arguments to a function, specifying argument data types – Illustration with example programs and problem solving through programs; Function prototypes, Functions returning non integers; Storage classes – Automatic, External, Static and Register variables, Scope rules, Header files, Block structure; Recursion in C – writing recursive programs and problem solving, The C Preprocessor

UNIT-III

Definition and array processing, passing arrays to a function, multidimensional arrays, Arrays and Strings; POINTERS – pointers and addresses, pointer declaration, pointers and function arguments – passing pointers to a function, Pointers and one dimensional arrays; Address arithmetic – operations on pointers, character pointers and functions; Pointer arrays/arrays of pointers, pointers to pointers, initialization of pointer arrays, pointers and multidimensional arrays; Command line arguments, Pointers to functions, passing functions to other functions.

UNIT-IV

Structures and Unions – Basics of structures, processing of structures, user defined data types (typedef), Structures and Pointers, Structures and functions – passing structures to a function, Arrays of structures, Pointers to structures, Self-referential structures, Table lookup, UNIONS. writing programs and problem solving with structure and union

UNIT-V

Input and output – Standard input and output, Formatted output – printf, Variable length argument, Formatted input - scanf; Data files – opening and closing data file, creating a data file, processing a data file, file access, unformatted data files, miscellaneous function in C; Low Level programming – Register variables, Bitwise operations, Bit fields, Enumeration, Commands Line arguments/parameters, Library functions, Macros, The C preprocessor.

Course outcome: After successful completion of the course, the students will be able to

1. *Learn the concepts of Programming in C*
2. *Learn thoroughly the building blocks of C programming language*
3. *Write and implement C programs and solve problems through programming*
4. *Learn the Concept of pointers and files in C & Programming with C.*
5. *Design and implement programs using pointers and files in C.*

Text Books:

1. Brian W. Kernighan and Denis M. Ritchie, **The C Programming Language**, Pearson Education India; 2nd edition, 2015.
2. Byron S Gottfried, **Programming with C**, McGraw Hill Education; 4th Edition, 2018.
3. E Balagurusamy, **Programming in ANSI C**, McGraw Hill Education; Eighth edition, 2019
4. V. Rajaraman, **Computer Programming in C**, PHI Learning Private Limited; Second edition, 2019

Reference Books:

1. Yashavant P. Kanetkar, **Let Us C**, BPB; 16th edition, 2017.
2. Kamthane, **Programming in C**, Pearson Education India; 3rd edition, 2015.

LAB: PART B: Programming with C (Practical): 30 Hours. (Practical /Project/Field work): Total marks: 30

Pass marks: 12

This part provides the practical knowledge of Programming with C.

Course Objectives:

1. *To explain design and implementation of C programs*

2. *To explain writing and executing programs with control structures and functions*
3. *To explain writing and executing programs with pointers in C*
4. *To explain writing and executing programs with structures and unions*
5. *To explain writing and executing programs with files in C*

Problem solving of various nature by implementing programs in C Programming languages based on unit wise contents of the theory paper Programming with C**.** Following are some programming tasks for laboratory programming assignments but the assignments are not limited to these only.

List of laboratory programming assignments (*not limited to these*)***:***

1. Write a program to
   a) Produce ASCII equivalent of given number
   b) Find the divisor or factorial of a given number.
   c) Evaluate the following algebraic expressions after reading necessary values from the user (ax+b)/ (ax-b) -2.5 log x-cos 30+|x2-y2|+sqrt (2xy) - (x5+10x4+8 x 3+4x+2)
   d) Find sum of a geometric series
   e) Cipher a string
   f) Check whether a given string follows English capitalization rules
   g) Find sum of the following series 1+ ½ + 1/3 +_____+1/20
   h) Search whether a given substring exists in an input string or not and then delete this string from the input string.
2. Write a recursive program for tower of Hanoi problem
3. The Fibonacci sequence of numbers is 1, 1, 2, 3, 5, 8……. Based on the recurrence relation F(*n*)=F(*n*-1)+F(*n*-2) for n>2 Write a recursive program to print the first *n* Fibonacci number
4. Write a menu driven program for matrices to do the following operation depending on whether the operation requires one or two matrices
   a) Addition of two matrices
   b) Subtraction of two matrices
   c) Finding upper and lower triangular matrices
   d) Trace of a matrix
   e) Transpose of a matrix
   f) Check of matrix symmetry
   g) Product of two matrices.
5. Write a program that takes two operands and one operator from the user perform the operation and then print the answer

7. Write functions to add, subtract, multiply and divide two complex numbers (x+iy) and (a+ib) Also write the main program.

8. Write a menu driven program for searching and sorting with following options:

   a) Searching (1) Linear searching (2) Binary searching

b) Sorting (1) Insertion sort (2) Selection sort

9. Write a program to copy one file to another, use command line arguments.
10. Write a program to mask some bit of a number (using bit operations)
11. An array of records contains information of managers and workers of a company. Print all the data of managers and workers in separate files.

Semester	**: II**
Course Type	**: SEC**
Course Code	**: CSCSEC151**
Name of the Course	**: Python Programming**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 30**
Total Marks	**: 100**
End Semester Marks	**: 80 [50 (Theory) + 30 (Lab)]**
Internal Marks	**: 20**

Course Objectives:

1. *Familiarize students with the basics of Python programming language, including syntax, variables, data types, and control structures syntax, variables, data types, and control structures.*
2. *Introduce students to fundamental programming concepts such as variables, data types, operators, conditional statements, loops, and functions, within the context of Python.*
3. *Explore various data structures in Python, such as lists, tuples, dictionaries, and sets, and develop the skills to manipulate, access, and organize data using these structures.*
4. *Teach students how to read from and write to files using Python, including text files and CSV files, enabling them to handle data stored in external files.*
5. *Develop skills in handling errors and exceptions in Python programs, allowing for graceful error handling and robustness.*

UNIT-I

Introduction: Basic Elements of Python, Python character set, Python tokens (keyword, identifier, literal, operator, punctuator), variables, concept of l-value and r-value, use of comments, Knowledge of data types: Number(integer, floating point, complex), boolean, sequence(string, list, tuple), None, Mapping(dictionary), mutable and immutable data types.Operators: arithmetic operators, relational operators, logical operators, assignment operators, augmented assignment operators, identity operators (is, is not), membership operators (in not in)Expressions, statement, type conversion, and input/output: precedence of operators, expression, and evaluation of an expression, type-conversion (explicit and implicit conversion), accepting data as input from the console and displaying output.

UNIT-II

Flow of Control: introduction, use of indentation, sequential flow, conditional and iterative flow; Conditional statements: if, if-else, if-elif-else, flowcharts, simple programs: e.g.: