Semester			Course Code	Course Title	Category
Semester –II			ESC103	Programming for Problem Solving (Theory & Lab.)	Engineering Science Course
Scheme and Credits				**Course contents**	
L	**T**	**P**	**Credits**	[The lab component should have one hour of tutorial followed or preceded by laboratory assignments.]	
3	**0**	**4**	**5**		

(i) Programming for Problem Solving ([L : 3; T:0; P : 0 (3 credits)] [contact hrs : 40]
Detailed contents
Unit 1 Introduction to Programming (4 lectures)
Introduction to components of a computer system (disks, memory, processor, where a program is stored and executed, operating system, compilers etc.) - (1 lecture). Idea of Algorithm: steps to solve logical and numerical problems. Representation of Algorithm: Flowchart/Pseudocode with examples. (1 lecture)
From algorithms to programs; source code, variables (with data types) variables and memory locations, Syntax and Logical Errors in compilation, object and executable code- (2 lectures) Unit 2:Arithmetic expressions and precedence (2 lectures)

Unit 2:Conditional Branching and Loops (6 lectures)
Writing and evaluation of conditionals and consequent branching (3 lectures)
Iteration and loops (3 lectures)

Unit 3 Arrays (6 lectures)
Arrays (1-D, 2-D), Character arrays and Strings

Unit 4 Basic Algorithms (6 lectures)
Searching, Basic Sorting Algorithms (Bubble, Insertion and Selection), Finding roots of equations, notion of order of complexity through example programs (no formal definition required)

Unit 5 Function (5 lectures)
Functions (including using built in libraries), Parameter passing in functions, call by value, Passing arrays to functions: idea of call by reference

Unit 6 Recursion (4 -5 lectures)
Recursion, as a different way of solving problems. Example programs, such as Finding Factorial, Fibonacci series, Ackerman function etc. Quick sort or Merge sort.

Unit 7 Structure (4 lectures)
Structures, Defining structures and Array of Structures

Unit 8 Pointers (2 lectures)
Idea of pointers, Defining pointers, Use of Pointers in self-referential structures, notion of linked list (no implementation)

Unit 9 File handling (only if time is available, otherwise should be done as part of the lab)

Suggested Text Books
(i) Byron Gottfried, Schaum's Outline of Programming with C, McGraw-Hill
(ii) E. Balaguruswamy, Programming in ANSI C, Tata McGraw-Hill

Suggested Reference Books
(i) Brian W. Kernighan and Dennis M. Ritchie, The C Programming Language, Prentice Hall of India

Course Outcomes
The student will learn
To formulate simple algorithms for arithmetic and logical problems. To translate the algorithms to programs (in C language).
To test and execute the programs and correct syntax and logical errors.

To implement conditional branching, iteration and recursion.
To decompose a problem into functions and synthesize a complete program using divide and conquer approach.
To use arrays, pointers and structures to formulate algorithms and programs.
To apply programming to solve matrix addition and multiplication problems and searching and sorting problems.
To apply programming to solve simple numerical method problems, namely rot finding of function, differentiation of function and simple integration.

(ii) Laboratory - Programming for Problem Solving [L : 0; T:0 ; P : 4 (2 credits)]
[The laboratory should be preceded or followed by a tutorial to explain the approach or algorithm to be implemented for the problem given.]

Tutorial 1: Problem solving using computers:
Lab1: Familiarization with programming environment

Tutorial 2: Variable types and type conversions:
Lab 2: Simple computational problems using arithmetic expressions

Tutorial 3: Branching and logical expressions:
Lab 3: Problems involving if-then-else structures

Tutorial 4: Loops, while and for loops:
Lab 4: Iterative problems e.g., sum of series

Tutorial 5: 1D Arrays: searching, sorting:
Lab 5: 1D Array manipulation

Tutorial 6: 2D arrays and Strings
Lab 6: Matrix problems, String operations

Tutorial 7: Functions, call by value:
Lab 7: Simple functions

Tutorial 8 &9: Numerical methods (Root finding, numerical differentiation, numerical integration):
Lab 8 and 9: Programming for solving Numerical methods problems

Tutorial 10: Recursion, structure of recursive calls
Lab 10: Recursive functions

Tutorial 11: Pointers, structures and dynamic memory allocation
Lab 11: Pointers and structures

Tutorial 12: File handling:
Lab 12: File operations

Laboratory Outcomes
To formulate the algorithms for simple problems
To translate given algorithms to a working and correct program To be able to correct syntax errors as reported by the compilers
To be able to identify and correct logical errors encountered at run time
To be able to write iterative as well as recursive programs
To be able to represent data in arrays, strings and structures and manipulate them through a program
To be able to declare pointers of different types and use them in defining self-referential structures.
To be able to create, read and write to and from simple text files.