PROGRAMMING FOR PROBLEM SOLVING LAB

I B. TECH- II SEMESTER								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIE	SEE	Total
A4CS02	ESC	-	-	4	2	30	70	100

COURSE OBJECTIVES:

1. To understand how to formulate the algorithms for simple problems
2. To be able to translate given algorithms to a working and correct program
3. To make them understand how to correct syntax errors as reported by the compilers
4. To be able to identify and correct logical errors encountered at run time
5. To understand how to write iterative as well as recursive programs
6. To enable them to represent data in arrays, strings and structures
7. To impart the knowledge of declare pointers of different types and their usage.
8. To understand how to create, read and write to and from simple text files.

COURSE OUTCOMES:

At the end of the course, student will be able to

1. Formulate the algorithms for simple problems
2. Translate given algorithms to a working and correct program
3. Correct syntax errors as reported by the compilers
4. Identify and correct logical errors encountered at run time
5. Write iterative as well as recursive programs
6. Represent data in arrays, strings and structures and manipulate them through a program
7. Declare pointers of different types and use them in defining self-referential structures.
8. Create, read and write to and from simple text files.

LIST OF EXPERIMENTS

WEEK-1 INTRODUCTION TO LINUX COMMANDS

1. Basic Linux commands
2. Write a C program to use printf() and scanf() functions
3. Write C programs to implement basic arithmetic operations – sum, average, product, difference, quotient and remainder of given numbers etc.

WEEK-2 OPERATORS AND EVALUATION OF EXPRESSIONS

1. Write a C program to check whether a number is even or odd using ternary operator.
2. Write a C program to perform the addition of two numbers without using +operator.
3. Write a C program to evaluate the arithmetic expression ((a + b / c * d - e) * (f - g)). Read the values a, b, c, d, e, f, g from the standard input device.
4. Write a C program to find the sum of individual digits of a 3 digit number.
5. Write a C program to read the values of x and y and print the results of the following expressions in one line:

   (x + y) / (x -y)

   (x + y)(x - y)

WEEK-3 CONDITIONAL STATEMENTS

1. Write a C program to find largest and smallest of given numbers.
2. Write a C program to find roots of a quadratic equation.
3. Write a C program which takes two integer operands and one operator form the user(+,-,*,/,% use switch)

WEEK-4 LOOPING STATEMENTS

1. Write a C program to find Sum of individual digits of given integer
2. Write a C program to generate first n terms of Fibonacci series
3. Write a C program to generate prime numbers between 1 and n

WEEK-5 LOOPING STATEMENTS

1. Write a C Program to find the Sum of Series SUM=1-x2/2! +x4/4!-x6/6!+x8/8!-x10/10!
2. Write a C program to generate Pascal's triangle.
3. Write a C program to generate pyramid of numbers.

```
1
                    1    3    1
               1    3    5    3    1
```

WEEK-6 **ARRAYS**

1. Write a C Program to implement following sorting methods
   Bubble sort
   Selection sort
   Insertion sort
2. Write a C program to find largest and smallest number in a list of integers

WEEK-7 **ARRAYS**

3. Write a C program
   To add two matrices
   To multiply two matrices
4. Write a C program to find Transpose of a given matrix

WEEK-8 **FUNCTIONS**

5. Write a C program to find the factorial of a given integer using functions
6. Write a C program to find GCD of given integers using functions
7. Write a C Program to find the power of a given number using functions

WEEK-9 **RECURSION**

1. Write a C Program to find binary equivalent of a given decimal number using recursive functions.
2. Write a C Program to print Fibonacci sequence using recursive functions.
3. Write a C Program to find LCM of 3 given numbers using recursive functions

WEEK-10 **STRINGS**

1. Write a C program using functions to
2. Insert a sub string into a given main string from a given position
3. Delete n characters from a given position in a string
4. Write a C program to determine if given string is palindrome or not

WEEK-11 **POINTERS AND STRUCTURES**

1. Write a C program to print 2-D array using pointers
2. Write a C program to allocate memory dynamically using memory allocation functions (malloc, calloc, realloc, free)
3. Write a C Program using functions to
4. Reading a complex number
5. Writing a complex number
6. Add two complex numbers
7. Multiply two complex numbers

Note: represent complex number using structure.

WEEK-12 **FILES**

1. Write a C program to copy one file to other
2. Write a C program to copy one file to other
3. Write a C Program to merge two files into a third file

TEXT BOOKS:

1. Yashavant Kanetkar, “Let Us C”, BPB Publications, New Delhi, 13th Edition, 2012.
2. Oualline Steve, “Practical C Programming”, O‘Reilly Media, 3rd Edition, 1997.

REFERENCE BOOKS:

1. King KN, “C Programming: A Modern Approach”, Atlantic Publishers, 2nd Edition, 2015.
2. Kochan Stephen G, “Programming in C: A Complete Introduction to the C Programming Language”, Sam‘s Publishers, 3rd Edition, 2004.
3. Linden Peter V, “Expert C Programming: Deep C Secrets”, Pearson India, 1st Edition, 1994.

WEB REFERENCES:

1. http://www.sanfoundry.com/c-programming-examples
2. http://www.geeksforgeeks.org/c
3. http://www.cprogramming.com/tutorial/c
4. http://www.cs.princeton.edu