19CS103 PROGRAMMING FOR PROBLEM SOLVING - I

Hours Per Week :

L	T	P	C
3	-	4	5

Total Hours :

L	T	P
45	-	60

WA/RA	SSH/HSH	CS	SA	S	BS
5	30	5	20	5	5

Source
http://www.trytoprogram.com/images

PRE-REQUISITE COURSES: NIL

COURSE DESCRIPTION AND OBJECTIVES:

This course is aimed to impart knowledge on basic concepts of C programming language and problem solving through programming. It covers basic structure of C program, data types, operators, decision making statements, loops, functions, static data structures. At the end of this course students will be able to design, implement, test and debug modular C programs.

COURSE OUTCOMES:

Upon completion of the course, the student will be able to achieve the following outcomes:

COs	Course Outcomes	POs
1	Understanding of how to write simple, but complete, C programs.	3
2	Identification of suitable data type operands and design of expressions having right precedence.	2
3	Application of decision making and iterative features of C Programming language effectively.	1
4	Selection of problem specific data structures and suitable accessing methods.	2
5	Design and development of non- recursive and recursive functions and their usage to build large modular programs.	3
6	Development of C programs that are understandable, debuggable, maintainable and more likely to work correctly in the first attempt.	3

SKILLS:

- ✓ Analysis of the problem to be solved.
- ✓ Design of algorithm/solution for a given problem.
- ✓ Identification of suitable data types for operands.
- ✓ Application of suitable control statements for decision making.
- ✓ Design of non-recursive and recursive functions to perform different tasks.
- ✓ Selection of static or dynamic data structures for a given problem and manipulation of data items.
- ✓ Development of C programs that are understandable, debuggable, maintainable and more likely to work correctly in the first attempt.

ACTIVITIES:

- *Analysis of a given problem.*
- *Design of algorithm/ solution.*
- *Implementation (coding and unit testing) of algorithm.*
- *System testing*

UNIT - I **L- 9**

INTRODUCTION TO ALGORITHMS AND PROGRAMMING LANGUAGES: Basics of algorithms, Flow charts, Generations of programming languages.

Introduction to C: Structure of a C program; pre-processor statement, Inline comments, Variable declaration statement, Executable statement; C Tokens: C Character set, Identifiers and Keywords, Type Qualifiers and Type Modifiers, Variables and Constants, Punctuations, and Operators.

UNIT - II **L- 9**

DATA TYPES AND OPERATORS: Basic data types; Storage classes; scope of a variable; Formatted I/O; Reading and writing characters; **Operators:** Assignment, Arithmetic, Relational, Logical, Bitwise, Ternary, Address, Indirection, Sizeof, Dot, Arrow, Parentheses operators; Expressions: Operator precedence, Associative rules.

UNIT - III **L- 9**

CONTROL STATEMENTS: Introduction to category of control statements; Conditional branching statements: if, if - else, nested-if, if – else ladder; Switch; Iterative statements: for, while, do - while, Nested loops; Break; Jump; goto and Continue.

UNIT - IV **L- 9**

ARRAYS: Introduction; Types of arrays; Single dimensional array: Declaration, Initialization, Usage, Reading, Writing, Accessing, Memory Representation, Operations; Multidimensional arrays.

UNIT - V **L- 9**

FUNCTIONS: User-defined functions; Function declaration: Definition, Header of a function, Body of a function, Function invocation; Call by value; Call by address; Passing arrays to functions; Command line arguments; Recursion; Library Functions.

LABORATORY EXPERIMENTS

LIST OF EXPERIMENTS **Total hours-30**

Experiment 1:

(a) Write a C program to display a simple text on the standard output device using puts ().

(b) Every character holds an ASCII value (an integer number in the range of 0 to 255) rather than that character itself, which is referred to as ASCII value. Likewise, for a given input whether it is character or digit or special character or lower case or upper case letter, find corresponding ASCII value.

Example: ASCII value of 'A' is 65.

Experiment 2:

(a) For the given Basic salary, compute DA, HRA and PF using the following criteria and find out the Net Salary of an Employee by deducting PF and IT.

DA = (Basic salary *25)/1000

HRA= (Basic salary * 15)/100

Gross salary = Basic salary + DA + HRA

PF = Gross salary * 10/100

IT= Gross salary * 10/100

Net Salary = Basic Salary + DA + HRA – (PF + IT)

(b) Write a C program to swap the two integers with and without using additional variable.

Example: Before swapping values of a =4, and b = 5 and after swapping a = 5, and b = 4.

Experiment 3:

(a) Write a C program to check whether a given character is a vowel or consonant.

Hint: Read input from the user, and check whether it is an alphabet or not. If it is an alphabet, then check whether it is a vowel or a consonant. Otherwise display it is not an alphabet.

(b) The marks obtained by a student in '*n'* different subjects are given as an input by the user. Write a program that calculates the average marks of given 'n subjects and display the grade. The student gets a grade as per the following rules:

Average	Grade
90-100	O
80-89	E
70-79	A
60-69	B
50-59	C
<50	F

Experiment 4:

(a) Write a C program to find HCF and LCM of the given two numbers.

Hint: Highest Common Factor (HCF) is also known as the greatest common divisor (GCD).

Example: HCF of the 9, 24 is 3, and LCM is 72

(b) Write a C Program to find the greatest factor of a given input other than itself.

Example: Consider, 30 is the given input, its greatest factor is 15.

Experiment 5:

(a) Write a C program to check whether a given number is an Armstrong number or not.

Hint: An Armstrong number is a number which is equal to the sum of digits raise to the power total number of digits in the number.

Example: Consider the Armstrong numbers are: $0(0^1)$, $1(1^1)$, $2(2^1)$, $3(3^1)$, $153(1^3+5^3+3^3=153)$, $370(3^3+7^3+0^3)$, $407(4^3+0^3+7^3)$, etc.

(b) Write a C Program to print the series of prime numbers in the given range.

Hint: The given number is prime if it is divisible only by one and itself.

Example:if the range is 5 and 15, return 5, 11 and 13 as the series of prime numbers in the given range.

Experiment 6:

(a) Write a C Program to print Floyd triangle for the user given number of rows. If the user entered 4 rows, then the output follows:

```
1
2 3
4 5 6
7 8 9 10
```

(b) Write a C Program to print the * for the given number of times in a rows to form a diamond shape. For the User Input 5, the output is

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

(c) Write a C Program to print Pascal triangle for the given number of rows. If the user entered 5 rows, then the output follows:

```
        1
      1   1
    1   2   1
  1   3   3   1
1   4   6   4   1
```

Experiment 7:

(a) Write a C Program to check whether the given number is a palindrome or not.

Hint: To check whether a number is a palindrome or not, reverse the given number and compare the reversed number with the given number, if both are same then the number is palindrome otherwise not.

Example: Given Number = 121, Reversed number = 121. Hence, given number is palindrome.

(b) Write a C Program to calculate sum of the individual digits for the given number.

Hint: To find the sum of the digits of a given number, use modulus operator (%) to extract individual digits of a number and keep on adding them.

Example: Given number is 9875. Sum of the given number "9875" is 9+8+7+5 = 29

Experiment 8:

Write a program to search for a given number in the given list of numbers.

Example: Read set of numbers *L*={2,4,6,1}. Search whether 4 is present in the given list or not.

Experiment 9:

Write a program to perform the following operations on a given list of elements.

(a) Insert the given element at the beginning of the list and at the end of the list.

Example: The given list is *L*={1,2,3,8}. Insert '0' at the beginning of the list and at the end of the list. Hence the resultant list is *L*={0,1,2,3,8,0}

(b) Delete an element at the beginning of the list and at the end of the list.

Example: The given list is *L*={1,2,3,8}. Delete an element at the beginning of the list and at the end of the list. Hence the resultant list is *L*={2,3}

Experiment 10:

Write a C program to perform the following operations on a list.

(a) Find the maximum or the largest element in a given list.

(b) Find the minimum or the smallest element in a given list.

Hint: Choose one dimensional array data structure.

Experiment 11:

Write a C program for the following:

(a) Calculate and print the sum of the elements in a one dimensional array, keeping in mind that some of those integers may be quite large.

Input Format:

- The first line of the input consists of number of data items in the array.
- The next line contains *n* space-separated integers contained in the array and print the sum of the elements in the array.

Example:

Enter 4 integers: 1000000001 1000000002 1000000003 1000000004. The sum of the given list is: 4000000010

(b) Write a program to reverse the given list, of size *n.*

Example: If the list, *L=[1,2,3]*, after reversing it, the list should be, *L=[3,2,1]*

Experiment 12:

Write a C program to perform addition, subtraction, multiplication operations on the two given matrices using functions.

Experiment 13:

Consider the below code segment which allows local and global variables. Find the local and global variables in this code segment. Write the output of this code segment.

```
#include<stdio.h>
int i;
void main()
{
        int j=60;
        i=50;
        f(i,j);
        printf("i=%d j=%d ", i,j);
}
f(int x, int y)
{
        i=100;
        x=10;
        y=y+i;
}
```

Experiment 14:

(a) Write a C program to compute the factorial of a given number using recursion.

Hint: Factorial is represented using '!' and it is calculated as n! = n*(n-1)*(n-2)*…*3*2*1.*As a function factorial(n)=n*factorial(n-1). Note: 0!=1.*

(b) Write a C program to swap two numbers using call by value and call by reference.

Experiment 15:

Write a C program that accepts a decimal number and outputs the binary representation of that number using user defined function.

Hint: Use the available built in functions if necessary.

Example: Enter the decimal number: 5. The binary representation for 5 is: 101

TEXT BOOKS :

1. Behrouz A. Forouzan, Richard F.Gilberg, “Programming for Problem Solving”, 1st edition, Cengage, 2019.
2. Ajay Mittal, “Programming in C - A practical Approach”, 1st edition, Pearson Education, India, 2010.

REFERENCE BOOKS:

1. Reema Thareja, “Computer Fundamentals and Programming in C”, 1st edition, Oxford University Press India, 2013.
2. Herbert Schildt, “C: The Complete Reference”, 4th edition, Tata McGraw-Hill, 2017.
3. Byron S Gottfried, “Programming with C”, 4th edition, Tata McGraw-Hill, 2018.