19CS101 PROGRAMMING FOR PROBLEM SOLVING

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

Hours Per Week :

L	T	P	C
3	-	2	4

Total Hours :

L	T	P
45	-	30

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

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 and dynamic 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	Understand how to write simple, but complete, C programs.	1
2	Identify suitable data type operands and design of expressions having right precedence.	2
3	Apply decision making and iterative features of C programming language effectively.	1
4	Design and develop of non-recursive and recursive functions and their usage to build large modular programs.	3
5	Select problem specific static/dynamic data structures and suitable accessing methods.	2
6	Develop C programs that are understandable, debuggable, maintainable and more likely to work correctly in the first attempt.	3

SKILLS:

- ✓ *Analyse a given problem to be solved.*
- ✓ *Design algorithm/solution for a given problem.*
- ✓ *Identify suitable data types for operands.*
- ✓ *Apply suitable control statements for decision making.*
- ✓ *Design non-recursive and recursive functions to perform different tasks.*
- ✓ *Select static or dynamic data structures for a given problem and manipulation of data items.*
- ✓ *Develop 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.*
- *System testing.*
- *Implementation (coding and unit testing) of algorithm.*

UNIT - I **L - 9**

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.

Data Types: Basic data types; Storage classes; scope of a variable; Formatted I/O; Reading and writing characters;

UNIT - II **L - 9**

OPERATORS AND CONTROL STATEMENTS: Operators **-** assignment, arithmetic, relational, logical, bitwise, ternary, address, indirection, sizeof, dot, arrow, parentheses operators; Expressions - operator precedence, associative rules; Control statements - category of statements, selection, iteration, jump, label, expression and block.

UNIT - III **L - 9**

ARRAYS AND FUNCTIONS: Array - declaration, initialization, reading, writing, accessing and passing as a parameter to functions, 2D-arrays, multidimensional arrays; Function - declaration, prototype, definition, calling by value and call by address, standard library functions and recursive functions.

UNIT - IV **L - 9**

STRINGS AND POINTERS: Strings - declaration, string library functions, array of strings, command line arguments; Pointers - declaration, initializing pointers, multiple indirection, relationship between arrays and pointers; Dynamic memory allocation functions.

UNIT - V **L - 9**

STRUCTURES AND UNIONS: Structures - defining a structure, declaration of a structure objects, operations on structures; Pointers to a structure; Array of structures; Nested structures; Unions; Bit – Fields.

LABORATORY EXPERIMENTS

LIST OF EXPERIMENTS **TOTAL HOURS: 30**

EXPERIMENT 1:

(a) Write a C program to display a simple message 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 print Floyd triangle for the user given number of rows.

Example: If the user entered 4 rows, then the output is as 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.

Example: If the user input is 5, then the output is as follows:

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

EXPERIMENT 5:

(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.

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

Hint: To find the sum of the digits of 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 digits of given number "9875" is
9+8+7+5 = 29

EXPERIMENT 6:

(a) Write a program to search for a given number in the given set of numbers.

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

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

i. 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}

ii. 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 7:

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 to store the given list of data items.

EXPERIMENT 8:

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

EXPERIMENT 9:

(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 10:

(a) Write a C program to read string using gets() function and use puts() function to print the contents of the string.

(b) Write a C program to copy a given string into another string without using standard string handling library function **strcpy()**.

Hint: Read one string as an input and then with the help of loop copy the content of given string into the new string. If the storage space allocated to the new string is less than the given string, entire string will not be copied into the new string.

Example: consider storage space allocated to new string is 20 and given string length is 30. In this case, your program can only copy 20 characters from given string into the new string.

EXPERIMENT 11:

(a) Write a C program to reverse a string without using standard string handling library function. Do not use another array to store the reversed string.

Hint: If a user enters a string "hello", then on reversing it will be displayed as "olleh".

(b) Write a C program to find whether the given two strings are same or not.

Hint: User need to enter two strings *s1 and s2* and check whether the two strings are same or not. For example: s1=hello, s2=hello output: YES

EXPERIMENT 12:

Write a C program for the following:

Given a string S, consisting of uppercase and lowercase letters, change the case of each alphabet in the string. That is, all the uppercase letters should be converted to lowercase and all the lowercase letters should be converted to uppercase.

Input: Vignan University

Output: vIGNANuNIVERSITY

EXPERIMENT 13:

(a) Write a C program to access the integer elements of the array using pointers.

Hint: Declare a pointer variable and assign the base address of the array to it and print the values of an array using pointer variable.

(b) Declare a character array to hold the input string and declare a character pointer variable. Assign the character array base address to the pointer and then display the every element of the character array.

Hint: Increment the pointer in loop.

EXPERIMENT 14:

Write a C program to count the number of vowels and consonants in a string using pointers.

Hint: Use pointers to read the content of the string.

EXPERIMENT 15:

Create a jagged array [array of variable length lists] with no of rows and no of columns in each row as specified by the user

Hint: Use Dynamic memory allocation (malloc() or calloc())

Input:

Enter no of rows: 3

Enter no of columns in Row 1: 3

Enter no of columns in Row 2: 5

Enter no of columns in Row 3: 2

Enter the elements row wise:

8 6 5

8 4 6 9 7

9 2

Output:

8 6 5

8 4 6 9 7

9 2

EXPERIMENT 16:

Write a C program for the following:

Customer billing system is a structure, having customers_name, street_address, city, state, account_number, payment_status(paid/ not_paid), payment_date(current date/ due_date), and amount as members. In this example, payment_date is also structure includes month, day and year as members. So, every customer record can be considered as nested structure. Display the payment_status of each customer.

Hint: Use nested structure concept.

EXPERIMENT 17:

Write a C program for the following: Define a structure named 'Complex' consisting of two floating point members called "real and imaginary". Let c1 and c2 are two Complex variables; compute the sum of two variables.

TEXT BOOKS:

1. Behrouz A. Forouzan and 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. ReemaThareja, "Introduction to C Programming", 2nd edition, Oxford University Press India, 2015.
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.