L	T	P	C
3	-	2	4

21BC102 PROGRAMMING FOR PROBLEM SOLVING- II

Course Description and Objectives:

This course is aimed to impart knowledge on advanced concepts of C programming language and problem solving through programming. It covers strings, pointers, static and dynamic data structures, and also file manipulations. At the end of this course, students will be able to design, implement, test and debug complex programs using advanced features.

Course Outcomes:

The student will be able to:

- Identify and understand the working of key components of a computer system (hardware, software, firmware etc).
- Understand computing environment, how computers work and the strengths and limitations of computers.
- Identify and understand the various kinds of input-output devices and different types of storage media commonly associated with a computer
- Identify and understand the representation of numbers, alphabets and other characters in computer system
- Understand, analyze and implement software development tools like algorithm, pseudo codes and programming structure

Skills:

- Analysis of the problem to be solved.
- Selection of static or dynamic data structures for a given problem and manipulation of data items.
- Application of various file operations effectively in solving real world problems.
- Development of C programs that are understandable, debuggable, maintainable and more likely to work correctly in the first attempt.

Activities:

- Study, analyze and understand logical structure of a computer program, and different construct to develop a program in 'C' language
- Write small programs related to simple/ moderate mathematical, and logical problems in 'C'.
- Study, analyze and understand simple data structures, use of pointers, memory allocation and data handling through files in 'C'.
- Identify and understand the working of different operating systems like windows and Linux etc.

Syllabus

UNIT - I **12 Hours**

STRINGS: Character array, Reading string from the standard input device, displaying strings on the standard output device, Importance of terminating a string, Standard string library functions.

UNIT - II **12 Hours**

POINTERS: Declaration, Initialization, Multiple indirections, Pointer arithmetic, Relationship between arrays and pointers, Scaling up - array of arrays, array of pointers, pointer to a pointer and pointer to an array; Dynamic memory allocation functions.

UNIT - III **12 Hours**

STRUCTURES: Defining a structure, Declaring structure variable, Operations on structures, Pointers to structure - declaring pointer to a structure, accessing structure members using pointer; Array of structures, Nested structures, Passing structures to functions - passing each member of a structure as a separate argument, passing structure variable by value, passing structure variable by reference/address; Typedef and structures.

UNIT - IV **12 Hours**

UNIONS: Defining an union - declaring union variable, operations on union; Pointers to union – declaring pointer to union, accessing union members using pointer; Array of union, Nested union, Typedef and union, Enumerations, Bit-fields.

UNIT - V **12 Hours**

FILES: Introduction to files, Streams, I/O using streams – opening a stream, closing stream; Character input, Character output, File position indicator, End of file and errors, Line input and line output Formatted I/O, Block input and output, File type, Files and command line arguments.

LABORATORY EXPERIMENTS:

Experiment 1:

(a) Write a C program to convert the given text into uppercase text.

Hint: Read a line of text character – by – character and store the characters in a char-type array. Read input characters until end-of-line (EOL) character has been read.

If the character is uppercase ignore it, otherwise convert it into uppercase using the library function toupper ().

Example: Hello Vignan HELLO VIGNAN

Experiment 2:

(a) Write a C program to read string using gets () function and print the contents of the string. In this case, your program can only copy 20 characters from given string into the new string.

(b) Write a C program to concatenate two strings without using standard string handling library function **strcat ()**.

Experiment 3:

Write a C program to concatenate the characters of the two given strings alternatively. **Hint**: If the length of the two strings is equal then concatenate the two strings alternatively otherwise concatenate the remaining characters of the higher length string at the end. Concatenated string is different from the given two strings.

Example: If “hi” and “vignan” are two strings then the concatenated string is “hivignan”.

Experiment 4:

(a) Write a C program to reverse a string without using standard string handling library function and, 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”.

Experiment 5:

Write a C program to remove blank spaces in the given string.

Input: Hello world

Output: Helloworld

Hint: Read the input through command line arguments. Removal of spaces should be performed on the given string itself.

Experiment 6:
Write a C program for the following:
Given a string S consisting of uppercase and lowercase letters, change the case of each alphabet in this 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: vIGNAN uNIVERSITY
Experiment 7:
Write a C program to insert a given character at the beginning and end of the given string.
Hint: If the input string is "C program" and the given character to insert is "g".
Input: "C program"
Output: "gC programg"
Experiment 8:
Write a C Program to find the frequency of occurrence, of a given character in the given string. **Hint**: Read a string and a character to be checked. Then count how many times that the given character has been repeated in the given string.
Example: The given string is: Chinthu, find the frequency of the occurrence of character 'h' in the given string. The frequency of occurrence 'h' in the given string is 2.
Experiment 9:
Write a C program to insert a character in a specified location of the given string.
Hint: Traverse the string upto the specified location, move the remaining characters back by one position and insert the given character at the specified location.
Example: If given string is 'Vignan, insert a character at 1st location and the given character is 'c'. Then the expected output is 'cVignan'.
Experiment 10:
(a) Write a C program to access the 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.
Experiment 11:
Write a C program to implement the following: Define a structure named 'Complex' consisting of two floating point members called "real and imaginary". Let c1 and c2 are two Complex structure variables; compute the sum of two variables.
Experiment 12:
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 an array of structures. Display the payment status of each customer.
Hint: Use nested structure concept.
Experiment 13:
(a) Write a C program to count the number of characters, number of lines and number of words in a given file.
Hint: Open a text file in read mode and count number of characters, number of lines and number of words in that file.
(b) Write a C program store the data in a text file.
Hint: Open a text file in write mode and read name, roll no and marks of n number of students from user and store the above details in the text file.

TEXT BOOKS:

1. Ajay Mittal, "Programming in C - A practical Approach", 1st edition, Pearson Education Publishers, India, 2010.
2. Reema Thareja, "Introduction to C Programming", 2nd edition, Oxford University Press India, 2015.

REFERENCE BOOKS:

1. Behrouz A. Forouzan and Richard F.Gilberg, "Programming for Problem Solving". 1st edition, Cengage Publishers, 2019.
2. Byron S Gottfried, "Programming with C", 4th edition, Tata McGraw-Hill Publishers, 2018.
3. Herbert Schildt, "C: The Complete Reference", 4th edition, Tata McGraw-Hill, 2017