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

# 21CS103 PROGRAMMING FOR PROBLEM SOLVING - I

Hours Per Week :

| L | T | P | C |
|---|---|---|---|
| 3 | - | 4 | 5 |

Total Hours :

| L | T | P |
|---|---|---|
| 45 | - | 60 |

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

## SKILLS:

- ✓ Analysis of the problem to be solved.
- ✓ Design of algorithm/solution for a given problem.
- ✓ Identify suitable data types for operands.
- ✓ Apply suitable control statements for decision making.
- ✓ Design of non-recursive and recursive functions to perform different tasks.
- ✓ Select static 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.

**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 statements, executable statements; C Tokens - C character set, identifiers and keywords, type qualifiers, type modifiers, variables, 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 case; Iterative statements - for, while, do - while, nested loops; Jump statements - break, jump, goto and continue.

**UNIT - IV** **L- 9**

**ARRAYS: I**ntroduction; 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.

**ACTIVITIES:**

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

## LABORATORY EXPERIMENTS

### LIST OF EXPERIMENTS Total hours:60

**Experiment 1:**

**Floating Point Number**

Write a C program to read the input(N) as floating point number from the user and display the number in one, two and three digit floating point numbers. Example: If N = 12.345678, then the output is 12.3, 12.34,12.345.

**Experiment 2:**

**Handshake!!**

Before the outbreak of corona virus to the world, a meeting happened in a room in Wuhan. A person who attended that meeting had COVID-19 and no one in the room knew about it! So everyone started shaking hands with everyone else in the room as a gesture of respect and after meeting unfortunately every one got infected!

Given the fact that any two persons shake hand exactly once. Write a C program that finds the total number of handshakes?

**Experiment 3:**

Ramya is learning about units and tens digit in school. Her teacher gives a task to find the sum of the ones and tens digit of any given number. She feels too lazy to do this assignment and hence assigned the job to her toy robot.

**Example:** Given number N = 2456, number 6 is in ones position and the number 5 is at tens position. Sum of the digits at ones position and tens position in the given number is: 6 + 5 = 11.

Write a C program that finds the sum of the digits at ones and tens position in the given number.

**Experiment 4:**

Write a C program that finds the number of pieces of candy will be eaten by all the children together.

‘n’ children have got ‘m’ pieces of candy. They want to eat as much candy as they can, but each child must eat exactly the same amount of candy as any other child. Determine how many pieces of candy will be eaten by all the children together. Individual pieces of candy cannot be split.

**Experiment 5:**

Write a C program that finds the century of a given year.

**Hint:** The first century spans from the year 1 to 100, the second century spans from the year 101 to the year 200, etc.

**Experiment 6:**

Write a C program that finds whether the given number is Lucky number or not.

**Lucky Number:** Ticket numbers usually consist of an even number of digits. A ticket number is considered as a lucky if the sum of the first half of the digits is equal to the sum of the second half.

**Example:** • For n = 1230, the output should be true; • For n = 23917, the output should be false.

**Experiment 7:**

Write a C program that finds whether the given number is smith number or not.

Smith Number: A Smith number is a composite number, the sum of its digits is the sum of the digits of its prime factors obtained as a result of prime factorization (excluding 1). The first few such numbers are 4,22,27,58,85,94 and 121.

**Example:**

378 = 2 x 3 x 3 x 3 x 7

So, its prime factors are 2, 3, 3, 3 and 7

The sum of its digits is 3 + 7 +8 =18.

The sum of the digits of its factors is 2+3+3+3+7 = 18

**Experiment 8:**

Write a C program that finds whether the 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 of total number of digits in the number.

**Experiment 9:**

Write a C Program that prints the Floyd triangle in 'n' rows.

**Example:** If n = 4

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

**Experiment 10:**

Write a C program that prints ' * ' in a diamond shape.

**Hint:** Read number of rows from the user.

**Example:** If n = 5

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

**Experiment 11:**

Write a C program that checks 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.

**Experiment 12:**

Write a C Program that calculates 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.

**Experiment 13:**

Write a C program to find the maximal value of the XOR operations for all permutations of the integers from 'l' to 'r', inclusive.

Given two integers, 'l' and 'r', find the maximum value of a xor b, written as a ^ b, where a and b satisfy the following condition: l < a < b < r.

**Experiment 14:**

Several people are standing in a row and need to be divided into two teams. The first person goes into team 1, the second goes into team 2, the third goes into team 1 again, the fourth into team 2, and so on. You are given an array of positive integers (the weights of the people). Return an array of two integers, where the first element is the total weight of team 1, and the second element is the total weight of team 2 after the division is complete.

Write a C program that finds the weight of team 1 and team 2 after the division.

**Experiment 15:**

Write a C program that searchs for a given number (I) in the given list of 'n' numbers.

**Experiment 16:**

Write a C program that finds the maximal absolute difference between any two of the adjacent elements in the given list of integers,

**Example :** Array = [2, 4, 1, 0], the output should be= 3.

**Experiment 17:**

Ramu got statues of different sizes as a present from CodeMaster for his birthday, each statue is a non-negative integer (size). Since he likes to make things perfect, he wants to arrange them from smallest to largest so that each statue will be bigger than the previous one exactly by 1. He may need some additional statues to accomplish that task. Write a C program that finds the minimum number of additional statues needed.

**Example :** For statues = [6, 2, 3, 8], the output should be = 3.

Ramu needs statues of sizes 4, 5 and 7

**Experiment 18:**

Write a C program for the following:

Alice and Bob each created two different challenges. A reviewer rates the two challenges, awarding points on a scale from 1 to 100 for three categories: problem clarity, originality, and difficulty. The rating for Alice's challenge is a = (a[0], a[1], a[2]), and the rating for Bob's challenge is b = (b[0], b[1], b[2]).

The task is to find their comparison points by comparing a[0] with b[0], a[1] with b[1], and a[2] with b[2].

- If a[i] > b[i], then Alice is awarded 1 point.
- If a[i] < b[i], then Bob is awarded 1 point.
- If a[i] = b[i], then neither person receives a point.

Comparison points is the total points a person earned. Given a and b, determine their respective comparison points.

**Example :** a = [1, 2, 3] b = [3, 2, 1]

- For the elements in 0th position, Bob is awarded a point because a[0] < b[0] .
- For the elements in 1st position, no points are earned because a[1] == b[1],
- Finally, for the elements in 2nd position, Alice receives a point because a[2] > b[2].
  The result is '1 1' (Alice's score first and Bob's second).

**Experiment 19:**

Write a C program that performs the following operations on a given list of elements.

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}

**Experiment 20:**

Write a C program that performs the following operations on a given list of elements. 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 21:**

Write A C Program That Find The Pair of Adjacent Elements That Has The Largest Product in The Given List of Integers And Return That Product.

**Example :** Array = [3, 6, -2, -5, 7, 3], the output should be= 21.

7 and 3 produce the largest product.

**Experiment 22:**

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

(a) In a given list, find the maximum or largest element.

(b) In a given list, find the least or minimum element.

**Hint:** Choose one dimensional array data structure.

**Experiment 23:**

Write a C program that prints the sum of the elements in a one dimensional array, (keeping in mind that some of those integers may be quite large.)

**Hint:** Declear array as long long integer.

**Experiment 24:**

Write a C program for the following:

Consider an arithmetic expression of the form A#B=C. Check whether it is possible to replace # with one of the four signs: +, -, *, / to obtain a correct expression.

**Experiment 25:**

Write a C program that accepts a decimal number and display its binary representation using user defined functions.

**Experiment 26:**

Write a C program that checks whether the two arrays are equal or not.

The two arrays are called similar if one can be obtained from another by swapping at most one pair of elements in one of the array.

**Example:** -> For a = [1, 2, 3] and b = [1, 2, 3], the output should be areSimilar(a, b) = true.

The arrays are equal, no need to swap any elements.

-> For a = [1, 2, 3] and b = [2, 1, 3], the output should be areSimilar(a, b) = true.

We can obtain b from a by swapping 2 and 1 in b.

-> For a = [1, 2, 2] and b = [2, 1, 1], the output should be areSimilar(a, b) = false.

Any swap of any two elements either in a or in b won't make a and b equal.

**Experiment 27:**

Write a C program that flips the bits in the given number.

Flipping bits: You will be given a list of 32 bit unsigned integers. Flip all the bits (1 à 0 and 0 à 1) and return the result as an unsigned integer.

**Example:** n = 910

910= 10012. We're working with 32 bits, so:

000000000000000000000000010012=910

11111111111111111111111101102=429496728610

The result is 429496728610

**Experiment 28:**

Write a C program that checks whether the given 10 digit mobile number is a fancy number or not.

Fancy number

Read a 10 digit mobile number, our task is to check whether the number is fancy number or not. There are three different conditions for a fancy number. If at least one is true, then the number is fancy. These conditions are like below.

- A single number occurs three consecutive times, like 555
- Three consecutive numbers are either in increasing or decreasing order like 123 or 321.
- A single digit occurs four or more times in a number, like 8965499259, here 9 has occurred four times.

One example of fancy number is 9859009976, this is a fancy number as the third condition satisfies.

**TEXT BOOKS:**

1. Behrouz A. Forouzan, Richard F.Gilberg, "Programming for Problem Solving", 1st edition, Cengage publications, 2019.
2. Ajay Mittal, "Programming in C - A practical Approach", 1stedition, 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", 4thedition, Tata McGraw-Hill, 2017.
3. Byron S Gottfried, "Programming with C", 4th edition, Tata McGraw-Hill, 2018.