



## SC: 504 - Computational Lab - I

Home Assignment

Sem - I, M.Sc.(Scientific Computing)

Batch - 32

Deadline: On or before 28<sup>th</sup> October 2025 :10 AM

Mark: 150

### Important instructions:

1. The assignment should be **handwritten**. Handwriting should be clear, readable and include : program number and attempt number
2. There should be **no syntax or logical errors**. Programs with errors will not receive marks.
3. You may use **pf** and **sf** as short forms for **printf** and **scanf**. You may skip **#include <stdio.h>**. If other headers are used, they must be included properly.
4. Prefer **quality over quantity** — do not try to rush or complete without understanding.
5. You may use AI tools (like ChatGPT) to generate programs, but you must **understand every line** you write.
6. It's **all or nothing** — each program must be written three times, or it will not count.
7. There is a **connection between programs** — the logic, syntax, and concepts build progressively.

### Solve ALL of the following :

#### 1. Section I - Read and Print (5Q)

- i. Read two integers, and perform addition, subtraction and division without using a third variable (3)
- ii. Read two integers, and perform addition, subtraction and division with a third variable (3)
- iii. Swap two numbers using a third variable (3)
- iv. Swap two numbers without using a third variable (3)
- v. Get and display the size of int, float, double, and char using sizeof (3)

#### Section II - Arithmetics (3Q)

- i. Accept an integer  $x$  and compute:  $x^2$  and  $x^2 + 2x$  (3)
- ii. Accept two integers  $x, y$  and compute:  $x^3 + 3x^2 + 4x - y^3$ ,  $\sqrt{x^3 + 3x^2 + 4x - y^3}$  and  $\frac{\sqrt{x^3 + 3x^2 + 4x - y^3}}{2x^2}$  (Use math.h header file) (3)
- iii. Accept the coefficients a, b and c of a quadratic equation and find its roots (3)

#### Section III - Conditional statements (12Q)

- i. Read three integers and find the largest among them. (3)
- ii. Read an integer and determine whether it is odd or even also check if it is positive, negative, or zero. (3)

- iii. Determine if a given year is a leap year. (3)
- iv. Read three angles of a triangle and check whether the triangle is valid. A triangle is valid if: Each angle is greater than  $0^\circ$  and sum of all three angles is  $180^\circ$ . (3)
- v. Write a program that reads three positive numbers  $a$ ,  $b$ , and  $c$  which represent the lengths of the sides of a triangle. Check if they form a valid triangle and categorize it (*equilateral*, *isosceles*, *scalene*). (3)
  - $a + b > c$ ,  $a + c > b$ , and  $b + c > a$  imply a **valid triangle**. item
    - $a = b = c \Rightarrow$  equilateral
    - $a = b$  or  $a = c$  or  $b = c \Rightarrow$  isosceles
    - $a \neq b \neq c \Rightarrow$  scalene
- vi. Input week number (1–7) and print corresponding weekday name. (3)
- vii. Input month number and display the number of days in the month (consider leap years for February). (3)
- viii. Read two integers, compute their quotient and remainder if the second is nonzero (3)
- ix. Read a integer - if it is divisible by 5 and 7 print - **Hello world** else print - **Bye**. (3)
- x. Input two numbers, find the maximum using the ternary operator. (3)
- xi. Use a ternary operator to find if a number is even or odd. (3)
- xii. Input a number (1–12) and display the name of the month via switch. (3)

#### Section IV - Loops (15Q)

- i. Print all even numbers between 1 and  $n$  (user input) using a **for loop**. (3)
- ii. Print all even numbers between 1 and  $n$  (user input) using a **while loop**. (3)
- iii. Print the first  $n$  terms of the Fibonacci series using a for loop. (3)
- iv. Calculate and print the sum of the series  $1 + 1/2 + 1/3 + \dots + 1/n$  using a for loop (3)
- v. Sum all integers from 1 to  $n$  using a while loop. (3)
- vi. Print a multiplication table for a given number using a while loop. (3)
- vii. Count and display the number of digits in a number using a while loop. (3)
- viii. Calculate the factorial of a number using a while loop. (3)
- ix. Accept an integer from user and reverse this number (3)
- x. Test whether a given integer is a palindrome. (3)
- xi. Calculate the average of positive numbers entered by the user. (3)
- xii. Skip printing odd numbers in a loop using continue. (3)
- xiii. Print multiplication tables from 1 to 10 using nested for loops. (3)
- xiv. Accept a number from user and check if it is prime or composite (3)
- xv. Accept a number  $n$  from user and test if all number print all prime numbers between 2 and  $n$ . (3)

#### Section V - 1D and 2D arrays (15Q)

- i. Dynamically allocate and read an array of size  $n$ . Find the sum and average of all elements in this array.. (3)

- ii. Write a C program to read  $n$  integers into a dynamically allocated array. Then count the number of good pairs in this array. A pair  $(i, j)$  is called good if  $arr[i] == arr[j]$  and  $i < j$ . (3)
- iii. Dynamically allocate and read an array of size  $n$ . Find the maximum and minimum element in this array. (3)
- iv. Dynamically allocate and read an array of size  $n$ . Count the number of even and odd numbers in this array. (3)
- v. Dynamically allocate and read an array of size  $n$ . Print the array in reverse order. (3)
- vi. Dynamically allocate and read two arrays of size  $n$ . Read input into the first array from user. Copy the content of first array into the second array. (3)
- vii. Dynamically allocate and read an array of size  $n$ . Search for a given number in an array using linear search. (3)
- viii. Dynamically allocate and read an array of size  $n$ . Separate positive and negative numbers into two arrays. (3)
- ix. Dynamically allocate and read an array of size  $n$ . Check if an array is a palindrome. (3)
- x. Dynamically allocate and read an array of size  $n$ . Move all zeros to the end of an array without changing the order of non-zero elements. (3)
- xi. Dynamically allocate and read an array of size  $n$ . Find the difference between the sum of even-indexed and odd-indexed elements. (3)
- xii. Write a program to find the second largest and second smallest element in an array. (3)
- xiii. Write a program to find the sum of all elements in a 2D array. (3)
- xiv. Write a program to find the largest and smallest elements in a 2D array. (3)
- xv. Write a program to add and subtract two matrices of the same dimensions. (3)

## Section VI - Extra [optional]

The following are to be written once.

- i. Display all elements that occur more than once in an array. (0)
- ii. Write a program to rotate the array elements to the right by  $k$  positions.  
Example:  $[1,2,3,4,5] \rightarrow [5,1,2,3,4]$  for  $k=1$ . (0)
- iii. Given an array containing numbers from 1 to  $n$  with one number missing. (Hint: Use the formula  $n*(n+1)/2$  or sum comparison). (0)
- iv. Write a program to find all leaders in an array. An element is called a leader if it is greater than all elements to its right.  
Example:  $[16,17,4,3,5,2] \rightarrow$  Leaders are 17, 5, 2. (0)
- v. Write a program to find the transpose of a given matrix. (0)
- vi. Write a program to multiply two matrix. (0)
- vii. Write a program to print Pascal's triangle using nested loops. (0)

>>> This is not punishment — it's skill training. <<<