Programming in C Questions For students practice

UNIT 1

1. Define a computer system. What are its main components Also draw the block diagram?
2. Explain the difference between primary memory and secondary memory with examples.
3. What is the role of the CPU (Central Processing Unit) in a computer system?
4. List and briefly describe any four input/output (I/O) devices with brief explanations of each.
5. What is the purpose of storage devices in a computer system? Name any two storage devices.
6. Define an assembler. What is its role in computer programming?
7. Differentiate between a compiler and an interpreter. Explain any four differences
8. Explain the functions of a loader and a linker in the process of executing a program.
9. What is an algorithm? Why is it important in problem-solving?
10. Describe the purpose of a flowchart in algorithm representation.
11. Explain what a pseudo code is, and how it is used in programming.
12. What are the basic components of a C program?
13. Describe the steps involved in writing and executing a C program.
14. What is the difference between syntax errors and logical errors in C programming?
15. Define variables in C. How are they related to memory locations?
16. **Write a c program to find the sum and average of a three numbers provided by the user.**
17. **Explain printf and scanf functions with example.**
18. **Explain about standard input/output header files in C.**
19. **Write a C program to find the greatest among three numbers provided by the user.**
20. Write a program to swap the values of two variables with and without using the third variable.

UNIT 2:

Arithmetic Expressions and Precedence

1. What is operator precedence, and how does it affect the evaluation of an expression involving both multiplication and addition? Provide an example.
2. Explain how the relational operators (<, >, ==, etc.) are used in expressions. Provide an example with a numeric comparison.
3. Describe how implicit type conversion works in an expression involving both an integer and a floating-point number.
4. What is the difference between the logical AND (&&) and logical OR (||) operators? Provide an example to illustrate your answer.
5. Explain how the bitwise OR (|) operator works on numeric values. Provide a simple example with binary numbers.

6. How does the assignment operator (=) work in an expression involving arithmetic operations? Provide an example to explain your answer.
7. What role does associativity play when evaluating an expression with multiple operators of the same precedence? Illustrate with an example.
8. Describe how mixed operands (e.g., integer and character) are handled in an arithmetic expression. Provide an example.

Conditional Branching

9. How does the else if construct help in evaluating multiple conditions in an if-else structure? Provide a brief code example.s
10. What is the purpose of the default case in a switch statement? Give an example to explain its use.
11. Explain how nesting of if-else statements can be used to evaluate complex conditions. Provide an example.
12. What happens when there is no break statement in a switch case? Illustrate with a simple code.

Iteration and Loops

13. Explain the main difference between a for loop and a while loop. When would you prefer one over the other?
14. How does the continue statement alter the flow of a loop? Provide an example to explain its effect.
15. Describe a situation where a goto statement might be used in a loop. Why is its use generally discouraged?
16. Write a program to find whether a person is eligible for voting or not?
17. Write a C program to print the odd and even number from 1 to 10.

UNIT 3, 4, 5 questions:

Programming Questions (20)

1. **Write a C program to find the largest and smallest element in an array.**
2. **Write a program to reverse the elements of an integer array.**
3. **Write a C program to multiply two 3x3 matrices.**
4. **Write a program to transpose a 4x4 matrix.**
5. **Create a program that takes a string as input and counts the number of vowels and consonants in it.**
6. Write a program to check if two given strings are anagrams.
7. Define a structure for a Student with fields for name, roll number, and marks. Write a program to calculate and display the average marks of a class of students.
8. **Write a program that defines an array of structures for Employee data (ID, Name, Salary). Sort the array based on Salary in ascending order.**
9. Write a program to define an Enum for days of the week. Write a function that takes an integer (1-7) and prints the corresponding day using the Enum.
10. Implement the Linear Search algorithm to find a given element in an integer array.

11. **Write a program to sort an array of integers in descending order using Bubble Sort.**
12. Write a function to calculate the sum of elements in an array. Use Call by Value and Call by Reference to illustrate the difference.
13. **Write a recursive function to calculate the factorial of a number.**
14. **Write a recursive function to find the Fibonacci sequence up to a given term.**
15. **Write a recursive function to calculate the power of a number.**
16. Write a program to swap two numbers using pointers.
17. Write a program to calculate the length of a string using pointers.
18. Write a program that creates an array of pointers to strings (e.g., store names of 5 countries).
19. **Write a program to sort an array of strings using pointers.**
20. Define a structure for Product with fields for name, code, and price. Write a program to dynamically allocate memory for an array of products and take input for each product's details.
21. Write a program that defines a structure for an employee and demonstrates how to access and modify structure members using a pointer to the structure.
22. Write a program that passes an array to a function and finds the maximum and minimum values.
23. Write a program that passes a pointer to an array and calculates the sum of all elements.
24. **Write a program that takes a 3x3 matrix as input and prints its diagonal elements using pointer notation.**
25. Write a program that dynamically allocates memory for a 2D matrix using pointers and fills it with values.
26. Write a program that uses a structure and a union to store data and display memory consumption for each.
27. Create a union to store data for different types of variables (int, float, char). Write a program to initialize and display each type of data.
28. Define an enumerated data type for a set of traffic light signals. Write a function that takes an enum value and displays the meaning of the signal.
29. **Implement the recursive version of the Bubble Sort algorithm to sort an array of integers.**
30. Write a program that accepts multiple strings from the user, stores them in an array of pointers, and prints the longest string.
31. Create a structure for Product with fields like name, price, and quantity. Sort an array of Product structures by price using pointers.
32. Write a program that uses a function pointer to call different arithmetic functions (addition, subtraction, multiplication, division) based on user input.
33. **Write a program to find the minimum and maximum element of the array.**
34. **Write a recursive function to calculate the factorial of a given number.**
35. **Write a program to print Armstrong numbers from 1 to 100 using a function.**
36. **Write a program to implement strlen (), strcat (),strcpy () using the concept of Functions.**
37. **Write a program to swap two elements using the concept of pointers.**

- **Programs + Theory based questions**

1. Describe the memory layout of single-dimensional and multi-dimensional arrays.
2. Explain row-major and column-major order in multi-dimensional arrays with examples.
3. **Explain how elements in an array can be modified. Provide examples of both one-dimensional and two-dimensional arrays.**
4. **Explain how memory is allocated for multi-dimensional arrays and how we can access elements in them.**
5. **Write the steps to initialize and access elements in a 3x3 matrix.**
6. Differentiate between character arrays and string literals in C.
7. Explain common string functions such as strlen, strcpy, strcat, and strcmp.
8. Differentiate between struct and union. Provide examples of when each should be used.
9. Explain how an enumerated data type works in C. Provide an example.
10. Describe how arrays of structures are declared and initialized. Explain with an example.
11. **Describe the working of the Linear Search algorithm and discuss its time complexity.**
12. Explain the Bubble Sort algorithm with an example and discuss its time complexity.
13. **Explain the different types of functions in C with examples.**
14. **Discuss the importance of function prototypes in C.**
15. Describe the difference between Call by Value and Call by Reference with examples.
16. Explain how arrays are passed to functions and how modifications to array elements are reflected in the calling function.
17. **Define recursion and give examples of problems that can be solved using recursive functions.**
18. Explain how pointers are declared and initialized in C. Provide examples of pointer arithmetic.
19. Describe how pointers are passed to functions and how this affects the modification of variable values.
20. **Differentiate between a pointer array and an array of pointers with examples.**
21. **Explain how pointers to structures work in C. Provide an example of accessing structure members using pointers.**
22. Discuss the use of malloc and free for dynamic memory allocation in C.
23. Explain with examples the use of arrays and pointers in different real-life applications.
24. Compare Linear Search with Binary Search. When is each method preferable?
25. **Discuss the use of recursive functions in sorting algorithms like Quick Sort and in search algorithms like Binary Search.**