Preliminary 75 examination questions for final exam (*Computer Programming, Fall 2024*)

1. Write C++ program that prints hello world message.

2. Write C++ program that performs addition.

3. Write C++ program that creates variables with different data types.

4. Write C++ program that prints several string text and some in new line.

5. Write C++ program that compares two variables with each other.

6. Write C++ program that finds out if two variable values are same or not same by comparing.

7. Write C++ program that finds out if two variable values are same or not same by comparing using ternary operator.

8. Write C++ program that uses while loop to increment value of local variable.

9. Write C++ program that uses do while loop to increment value of local variable.

10. Write C++ program to demonstration increment operation with post and pre options.

11. Write C++ program to demonstration decrement operation with post and pre options.

12. Find Syntax errors from the following code snippet.

```
int main() {

    int counter = 1:
    do
        cout << counter << "  ";
        counter++;
    }
    while (counter <= 10)
```

13. Find Syntax errors from the following code snippet.

```
int number1 - 0;
int number2 = 0;

cout >> "Enter two integers to compare: ":
cin >> number1 >> numberTwo;
```

14. Find Logic errors from the following code snippet.

```
for(int i = 1; i < 5; i++) {

    num = 1 + rand() % 11;
    cout << num << " ";
}
```

15. Write complete C++ program that takes two input and performs addition and subtraction.

16. Write complete C++ program that takes two input and performs multiplication and division.

17. Write complete C++ program that demonstrates use of modulus (%) operator.

18. Write C++ code snippet for integer array with four values and print values of array with while loop.

19. Write C++ code snippet for floating point array with four values.

20. Write C++ code snippet that uses array in for loop.

21. Explain the purpose of variable in computer program.

22. What is function in computer programming.

23. What is considered as Syntax error in computer programming.

24. What is considered as Logic and runtime error in computer programming.

25. What are the common steps of creating the computer program. How C++ program creation is different from Java and PHP program build.

26. What is the purpose of a conditional statement in programming?

27. What is the difference between an "if" statement and an "if-else" statement? Provide an example for each.

28. Explain how the "else if" statement works in a chain of conditions. Can you give an example where it might be useful?

29. What does the "switch" statement do, and how is it different from a series of "if-else" statements?

30. What is a ternary operator, and how does it function as a shorthand for an "if-else" statement? Provide an example.

31. What will the following code print? Int x = 10; if(x < 5) cout << "Less than 5"; else if(x == 10) cout << "Equal to 10"; else cout << "Greater than 5";

32. What is a "nested if" statement? Provide an example and explain why it might be used.

33. In which situations would you prefer using a "switch" statement over multiple "if-else" conditions?

34. How do logical operators like && (AND) and || (OR) affect the evaluation of multiple conditions in an "if" statement? Provide an example.

35. What happens if the condition in an "if" statement is always false, and the "else" part is omitted? Will the code still execute? Explain.

36. What is a loop in programming, and why is it important for automating repetitive tasks?

37. What is the primary difference between a "for" loop and a "while" loop in terms of their structure and use cases?

38. Explain the concept of an "infinite loop." What could cause a loop to become infinite, and how can it be avoided?

39. What is a "nested loop"? In what scenarios might you use a loop inside another loop? Provide a conceptual explanation.

40. Describe how the "for" loop works in most programming languages. What are the three main parts of a typical "for" loop?

41. What is an "off-by-one" error in a loop, and how does it occur? How can you prevent this error when writing loops?

42. How does the "break" statement function within a loop? What effect does it have on the flow of the program?

43. What is the purpose of the "continue" statement in a loop, and how does it differ from the "break" statement?

44. In what ways can a "while" loop and a "do-while" loop be distinguished, especially in terms of loop execution?

45. What is the significance of the loop termination condition in both "for" and "while" loops? What can go wrong if this condition is not carefully handled?

46. Explain how loops can be used to iterate over collections (such as arrays or lists) in programming. Why is this a powerful technique?

47. What is the role of the "else" clause in a loop, if supported by the programming language? How does it differ from the "else" clause in conditional statements?

48. Describe how the "for-each" loop (or enhanced for loop) works. How does it simplify iterating over arrays or collections compared to a traditional "for" loop?

49. In a loop that iterates over a list, what considerations should be made when modifying the list (e.g., adding or removing elements) while it is being iterated over?

50. What are the performance implications of using loops, particularly nested loops, in terms of time complexity? How do you optimize loops to improve performance?

51. What is a constructor in object-oriented programming, and what is its purpose when creating an object of a class?

52. What is the difference between a default constructor and a parameterized constructor? Provide an example of each.

53. What is the role of a setter method in a class, and how does it contribute to data encapsulation?

54. How does a getter method work in a class, and why is it important for accessing private data members?

55. What is the main reason for using setter and getter methods instead of directly accessing object properties?

56. What is a struct in programming, and how does it differ from a class in terms of memory allocation and behavior?

57. Can a struct have a constructor? If so, how is it different from a class constructor? Explain with an example.

58. How are setters and getters typically used in combination to modify and retrieve values of private data members in a class? Provide a code example.

59. What are the advantages of using a struct instead of a class for simple data storage purposes, and what limitations might exist?

60. Explain the concept of object initialization with respect to constructors, and why it's important to initialize an object’s properties during object creation.

Variables:

61. What is a variable in programming, and why are variables important?

62. Explain the difference between local and global variables. Provide examples of when to use each.

63. What are the common data types used in variables, and how do they differ?

64. Describe the concept of variable scope and how it affects a program's behavior.

65. What are constants, and how do they differ from regular variables? Why are constants used in programming?

Functions:

66. What is a function, and what are its key components?

67. Explain the concept of function parameters and arguments. How are they used to make functions more flexible?

68. What is the difference between a return value and a side effect in the context of functions? Provide examples.

69. Compare and contrast recursive functions and iterative functions. When is recursion preferred?

70. What are anonymous functions (e.g., lambda functions)? In what scenarios are they typically used?

71. What is an array in programming, and how does it differ from other data structures like lists or dictionaries?

72. How can you access and modify elements in an array? Provide an example in a programming language of your choice.

73. Explain the concept of array indexing. What is the difference between zero-based and one-based indexing?

74. What are multidimensional arrays, and how are they used? Give an example of a 2D array and describe a scenario where it might be useful.

75. Discuss the advantages and limitations of arrays compared to other data structures, such as linked lists or hash tables.

~ ~ END OF DOCUMENT ~~