Spring 2025, MIS 102 – COMPUTER PROGRAMMING

Midterm Exam

姓名:_____ 學號:_____ 系級:_____

Part Ⅰ. [60 pts]

1. Evaluate the following C code. Which statement is **FALSE**?

```
const int * const ptr;
```

A. ptr is a pointer variable that points to an object of type int

B. The memory address saved in ptr CANNOT be modified.

C. The data value of the object that ptr points to can be modified.

D. The statement ptr = &y is invalid.

2. Assuming that t is an array and tPtr is a pointer to that array, which of the following statements is **INVALID**?

A. t == tPtr

B. t[0] == *tPtr

C. &t[2] = tPtr + 2

D. t[0] == tPtr[0]

3. Let's say there is a float array, float f[10], in the main memory. Its first element is at location 5002. A pointer vPtr has been initialized to point to f[0]. The value of vPtr+=4 is _____, and when vPtr points to f[5], we can use _____ to point back to f[2]?

A. 5010, vPtr -= 3

B. 5010, vPtr -= 6

C. 5018, vPtr -= 6

D. 5018, vPtr -= 3

4. In C, given the declaration int arr[3][4];, which option best describes the type of the array name arr?

A. A pointer to an array of 4 integers

B. A pointer to 3 integers

C. The same type as &arr[0][0]

D. A one-dimensional array that can traverse all elements

5. After using **malloc** to allocate memory, assume that p is a pointer to the beginning of the allocated block. Which statement is correct regarding p?

A. p + 1 moves the pointer by one byte

B. p + 1 moves the pointer by the size (in bytes) of its data type

C. Using p++ results in undefined behavior

D. Pointer arithmetic requires explicit type casting before addition

6. Regarding recursive functions, which statement is TRUE?

A. Recursive calls do not affect stack memory usage

B. Recursion is generally more efficient and uses less memory than loops

C. Each recursive call creates a new stack frame, which can lead to stack overflow

D. All recursive problems can be directly replaced by loops without affecting the program structure

7. What is the primary advantage of using the top-down design approach in structured programming?

A. Writing all code in a single function to simplify management

B. Facilitating problem decomposition, modular design, and easier maintenance

C. Forcing the use of global variables to increase execution speed

D. Eliminating loops and conditional statements from the program

8. In the context of multiple nested for loops, what is the best practice to avoid variable name conflicts and scope confusion?

A. Use the same loop variable for all nested loops

B. Fix the value of all variables at compile time to prevent conflicts

C. Define the variable in the outer loop only and reuse it in inner loops

D. Use separate, block-scoped loop variables for each level of nesting

9. After using malloc() to allocate memory, which statement is most accurate?

A. Memory allocated with malloc is automatically reclaimed by the system without manual intervention.

B. The allocated memory block is automatically initialized to zero.

C. It is necessary to use free to release the allocated memory to avoid memory leaks.

D. Once memory is allocated, there is no need to check if the returned pointer is NULL.

10. Given a double pointer declaration, int **ptr; which option best describes a common usage of such a pointer?

A. It points to the address of a single integer variable.

B. It points to the first element of an integer array.

C. It is used to reference an array of pointers or a dynamically allocated two-dimensional array.

D. It is exclusively used to store the return value of a function that returns an integer.

11. Which of the following best describes the primary advantage of modular programming?

A. It allows using global variables throughout the program.

B. It eliminates the need for function prototypes.

C. It helps in reusing, testing, and maintaining code easily.

D. It improves pointer arithmetic performance.

12. What is the main purpose of the **`const`** keyword when used with array parameters in a function?

A. It allows modifying the array within the function

B. It prevents the compiler from optimizing the code

C. It ensures that the function cannot modify the array contents

D. It makes the array act as a global variable

13. Which type of C variable retains its value between multiple calls of a function but is still only accessible inside that function?

A. Global variable

B. Static local variable

C. Extern variable

D. Auto variable

14. Which of the following situations would require using "pass by reference" rather than "pass by value"?

A. To return multiple values from a function

B. To prevent a function from modifying the input

C. When working only with constants

D. To improve code readability

15. Which of the following is **FALSE** about arrays in C?

A. Array names decay to pointers in function calls

B. C automatically checks array bounds at runtime

C. Arrays store elements in contiguous memory locations

D. Indexing starts from 0 in C

16. When a pointer is incremented using **`ptr++`**, what actually happens?

A. The pointer moves to the previous memory address

B. The pointer moves to the next byte in memory

C. The pointer moves ahead by the size of the data type it points to

D. The pointer's value is replaced with NULL

17. Which of the following loop types is **best suited** for reading unknown-length user input until a special value is detected?

A. do-while

B. for

C. while

D. recursive function

18. What is the main difference between **`char str[6] = "HELLO";`** and **`char *str = "HELLO";`**?

A. The first stores the string in heap, the second on the stack

B. The first is mutable, the second points to a read-only memory location

C. The first is a pointer, the second is an array

D. Both are functionally the same

19. What does the **`sizeof`** operator return when applied to a pointer?

A. Size of the data it points to

B. Size of the pointer itself

C. Always returns 4

D. The number of elements in an array

20. Which of the following best explains **why** a static local variable is initialized only once?

A. It belongs to the global memory space

B. It is stored on the stack frame

C. Its lifetime persists across function calls

D. It is reset each time the function runs

21. Which of the following expressions demonstrates pointer dereferencing?

A. &x

B. *x

C. x++

D. x = 10

22. Which logical operator in C short-circuits if the first operand evaluates to false?

A. ||

B. &&

C. ==

D. !=

23. Which of the following is a **reason to avoid using `goto`** in structured programming?

A. It makes loops faster

B. It increases readability

C. It causes unpredictable compilation

D. It leads to unstructured and hard-to-maintain code

24. Given the expression **`x = 3 + 4 * 2 > 10 && 5 == 5;`**, what is the final value of **`x`**?

A. 0

B. 1

C. 11

D. Syntax error

25. Which memory segment stores **local variables** in a function by default?

A. Heap

B. Stack

C. Data segment

D. Code segment

26. Which of the following statements about recursion is FALSE?

A. It must include a base case to prevent infinite calls

B. It can be replaced by iteration for better performance

C. Each recursive call uses a new stack frame

D. Recursive functions are always more efficient than loops

27. Which of the following is **TRUE** about **`void*`** (void pointer) in C?

A. It cannot store any memory address

B. It can be dereferenced without typecasting

C. It is used for pointers of unknown data types

D. It can only point to integers

28. What happens if you access an array element beyond its declared bounds in C?

A. The compiler throws an error

B. The runtime environment halts execution

C. Unpredictable behavior or memory corruption can occur

D. The out-of-bounds element is initialized to zero

29. What would happen if you place a `continue;` statement as the first line inside a **`for`** loop's body?

A. The loop condition will never be evaluated

B. The loop will skip all iterations

C. The increment step is still executed after **`continue;`**

D. The loop will result in an infinite loop

30. In function prototypes, which component ensures the compiler can perform type checking?

A. Function body

B. Variable scope

C. Parameter list and return type

D. The use of **`static`**