Spring 2025, MIS 102 – COMPUTER PROGRAMMING

Quiz 3

姓名:_____ 學號:_____ 系級:_____

1. [60 pts]

1. A function prototype does NOT have to _____.
(a) include parameter names
(b) terminate with a semicolon
(c) agree with the function definition
(d) match with all calls to the function

Ans. (a)

2. The following code:

```
double mySqrt( int x );
```

(a) defines a function called double which calculates square roots
(b) defines a function called mySqrt which takes an integer as an argument and returns a double
(c) defines a function called mySqrt which takes an argument of type x and returns a double
(d) defines a function called mySqrt which takes a double as an argument and returns an integer

Ans. (b)

3. Which of the following about global variables is FALSE?
(a) Variables declared outside of a block (in a {}) are often called global variables.
(b) Global variables can be accessed anywhere in the same program.
(c) When blocks are nested, and an identifier (variable name) in an outer block (global variables) has the same name as an identifier in an inner block(local variable), the identifier in the inner block is “hidden” until the inner block terminates.
(d) Global variables can be accessed in any functions and made changes to it.

Ans. (c)

4. Which statement about C functions is FALSE?
(a) Function call stack is used to store return values of function calls.
(b) If you cannot choose a concise name that expresses what a function does, it's possible that the function is attempting to perform too many diverse tasks.
(c) Each function should be limited to performing a single, well-defined task.
(d) "Information hiding" means that function A does not know how function B performs its tasks, which may lower errors in a software system.

Ans. (a)

5. Which of the following about main() is TRUE?
(a) Normally, main() has a “void” return data type.

(b) If you omit the return statement, 0 will be returned. Thus, it is encouraged to omit the return statement.
(c) A usual C program can have more than one main().
(d) There can be multiple parameters in main().

Ans. (d)

6. Which of the following statements is TRUE?
(a) In a C program, programmer-defined headers are enclosed in double quotes ("")
(b) If we would like to use functions we created in our C program, we have to create headers.
(c) Headers are used to define local variables.
(d) A built-in compiler header file is a file with extension .hf and can be included by using the #include preprocessor directive with double quotes ("").

Ans. (a)

7. Which of the following statements is TRUE?
(a) Headers are used to define functions. That is, the function prototype.
(b) Each standard library has a corresponding headers containing the function prototypes for all the functions in that library.
(c) A programmer-defined header file is a file with extension .hf and can be included by using the #include preprocessor directive.
(d) If we create functions in our C program, we have to create headers.

Ans. (b)

8. Which of the following statements is FALSE?
(a) Iteration is based on a repetition structure that terminates when the loop-continuation condition fails.
(b) Both iteration and recursion involve some form of repetition.
(c) Iteration may occur infinitely, so it is better to use recursion.
(d) Recursion achieves repetition through repeated function calls.

Ans. (c)

9. What does the static keyword do when applied to a local variable in a function?
(a) Makes it global
(b) Allocates it on the heap
(c) Limits access to only within that source file
(d) Preserves its value across multiple function calls

Ans. (d)

10. What is the primary function of the call stack in managing functions?
(A) Storing global variables
(B) Managing variable types
(C) Storing function definitions
(D) Keeping track of active function calls and their local data

Ans. (d)

11. Why is recursion generally less efficient than iteration in C for computing large factorial values?
(A) Because recursion does not support multiplication
(B) Due to repeated memory allocation for global variables

(C) Because each recursive call adds a new stack frame, increasing memory and execution overhead
(D) Because iterative code cannot be optimized

Ans. (c)

12. Which of the following Python practices most closely mirrors the use of header files (.h) in C for modularity and team collaboration?
(a) Using global variables to share data across modules
(b) Writing all functions in a single file
(c) Defining functions in a .py file and importing them into other modules using import
(d) Using __main__ to check if the script is being run directly

Ans. (c)

13. According to the KISS principle, which of the following is the BEST approach to function design?
(a) Combine multiple related tasks into one large function
(b) Design functions that perform a single, well-defined task
(c) Create complex logic in a single function to reduce file size
(d) Use global variables to simplify function interfaces

Ans. (b)

14. What role does the linker play in the compilation process of a C program?
(A) It checks syntax errors in the code
(B) It converts source code into object code
(C) It connects function calls in one file to their definitions in another file
(D) It compresses the final executable file

Ans. (c)

2. [40 pts]

Q1 (20 pts): Write a C program that reads a positive integer from the user and uses a single function named processNumber to do the following:

1. Reverse the digits of the input number
2. Calculate the square of the reversed number
3. Return the squared value to main, where it will be printed

---

Requirements:

- You must define only one custom function: int **processNumber(int n);**
- The input will always be a **positive** integer (no need to handle invalid input)
- The main function should handle user input and output
- The reversing and squaring logic should be inside processNumber

---

Here are some examples:

```
Example Input:
Enter a number: 123

Example Output:
Reversed number: 321
Square of reversed number: 103041

Example Input:
Enter a number: 50

Example Output:
Reversed number: 5
Square of reversed number: 25
```

```c
#include <stdio.h>

// Function prototype
int processNumber(int n);

int main() {
    int num, result;

    printf("Enter a number: ");
    scanf("%d", &num);

    result = processNumber(num);

    printf("Square of reversed number: %d\n", result);

    return 0;
}

// Function definition
int processNumber(int n) {
    int reversed = 0;

    // Reverse the digits
```

```
    while (n > 0) {
        reversed = reversed * 10 + (n % 10);
        n /= 10;
    }

    printf("Reversed number: %d\n", reversed);

    // Return the square of the reversed number
    return reversed * reversed;
}
```

Q2 (20 pts): Write a Python program that defines a function called is_palindrome_number(n) to determine whether a given positive integer n is a palindrome number.

A palindrome number is a number that reads the same forwards and backwards.
Examples:

- 121 is a palindrome
- 1331 is a palindrome
- 123 is not a palindrome

Requirements:
1. Define only one function: **is_palindrome_number(n)**
2. In the main program, prompt the user to input an integer, then call the function
3. Print "**Yes, it's a palindrome.**" if true, or "**No, it's not a palindrome.**" if false

---

Here are some examples:

```
Example Input:
Enter a positive integer: 121
Example Output:
Yes, it's a palindrome.
```

```
Example Input:
Enter a positive integer: 123
Example Output:
No, it's not a palindrome.
```

```
def is_palindrome_number(n):
    original = str(n)

    reversed_num = original[::-1]

    return original == reversed_num

# Main program
num = int(input("Enter a positive integer: "))

if is_palindrome_number(num):

    print("Yes, it's a palindrome.")

else:

    print("No, it's not a palindrome.")
```