Spring 2025, MIS 102 – COMPUTER PROGRAMMING

Quiz 4

姓名:__________________ 學號:__________________ 系級:__________________

1. [60 pts]

1. What does this declaration `int A[3] = {0}` do?

   a. Initializes only the first element to 0

   b. Leaves all elements uninitialized

   c. Initializes all three elements to 0

   d. Initializes the last element to 0

   ANS：C

2. What is true about passing an array to a function in C?

   a. The entire array is copied to the function

   b. Only the last element is passed

   c. The array name decays to a pointer to the first element

   d. It is passed by value by default

   ANS：C

3. Which of the following best explains why the use of `const` with array parameters improves function design?

   a. It allows arrays to be passed by value

   b. It enforces the array to be global

   c. It protects the original array from unintended modifications

d. It allows the compiler to allocate more memory

ANS : C

4. Which of the following best explains why arrays in C are passed to functions as pointers rather than by value?

   a. Because C does not support call-by-value semantics for arrays

   b. Because arrays are dynamically allocated in C

   c. To avoid the performance cost of copying large blocks of memory

   d. Because the array name is always treated as a global variable

   ANS : C

5. What happens if you try to modify a **`const int arr[]`** inside a function?

   a. It silently fails

   b. It compiles but throws a warning

   c. The program will crash at runtime

   d. It results in a compile-time error

   ANS : D

6. If **`sizeof(arr)`** is used inside a function that receives **`int arr[]`** as a parameter, what will it return?

   a. The total size of the array in bytes

   b. The number of elements in the array

   c. The size of the pointer to the first element

   d. Compilation error

   ANS : C

7. Which of the following is a valid declaration of a 2D array with 3 rows and 4 columns?

   a. `int arr[4][3];`

b. `int arr[3, 4];`

c. `int arr[3][4];`

d. `int arr(3, 4);`

ANS:C

8. If **`int arr[5] = {1, 2, 3, 4, 5};`** is declared in a function, where is this memory typically allocated?

a. Heap

b. Stack

c. Data segment

d. Code segment

ANS:B

9. What is the correct way to declare a function that takes a 2D array with 10 columns?

a. `void func(int arr[][]);`

b. `void func(int* arr);`

c. `void func(int arr[][10]);`

d. `void func(int arr[10][]);`

ANS:C

10. If a pointer is declared as **`int* const ptr`**, which of the following is **TRUE**?

a. The pointer cannot be dereferenced

b. The data pointed to cannot be modified

c. The pointer cannot be reassigned to point elsewhere

d. The pointer is automatically set to NULL

ANS:C

11. Which of the following statements about static arrays is **TRUE**?

a. They are destroyed at the end of each function call

b. They cannot be initialized

c. Their values persist between function calls

d. They use more memory than dynamic arrays

ANS : C

12. Suppose a function prototype is written as **`void process(int arr[10]);`**. Which of the following is TRUE?

a. The compiler ensures only arrays of size 10 can be passed

b. The parameter is still treated as int *arr

c. The array elements are copied to the function

d. This leads to undefined behavior unless the size is 10

ANS : B

13. Which of the following reasons best describes why C does not perform array bounds checking at runtime?

a. To increase security

b. Because C uses static memory only

c. To allow dynamic typing

d. To maximize performance and low-level memory control

ANS : D

14. Unless otherwise specified, entire arrays are passed ___________ and individual array elements are passed ___________.

(a) call-by-value, call-by-reference

(b) call-by-reference, call-by-value

(c) call-by-value, call-by-value

(d) call-by-reference, call-by-reference

ANS : B

15. In Python 3, suppose d = {"andy": 40, "tom": 45}, what happens when we try to retrieve a value using the expression d[“merry”]?

   a) Since “merry” is not a value in the set, Python raises a KeyError exception

   b) It is executed fine and no exception is raised, and it returns None

   c) Since “merry” is not a key in the set, Python raises a syntax error

   d) Since “merry” is not a key in the set, Python raises a KeyError exception

   ANS : D

2. [40 pts]

Q1 [C] (20 pts) : Suppose there are two arrays, one storing student IDs and the other storing student scores. You need to write a C program that prompts the user to input a student ID.The program should then find the corresponding index of that ID in the array of student IDs and retrieve the corresponding score from the array of scores. Finally, the program should output the student's score.

Hint: Assuming the student IDs and scores are as follows, please use these two pre-written arrays to write the program:

```
int studentIds[] = {101, 102, 103, 104, 105, 106, 107, 108, 109, 110};
int scores[] = {85, 92, 78, 88, 95, 80, 87, 90, 83, 91};
```

Here is an example of the program output:

```
Enter your student ID: 102
Your score is 92.
Enter your student ID: 110
Your score is 91.
Enter your student ID: 112
The student ID is invalid.
```

ANS :

```c
#include <stdio.h>

int main() {
    // Define the arrays for student IDs and scores
    int studentIds[] = {101, 102, 103, 104, 105, 106, 107, 108, 109, 110};
    int scores[] = {85, 92, 78, 88, 95, 80, 87, 90, 83, 91};

    // Prompt the user to enter a student ID
    printf("Enter your student ID: ");
    int inputId;
    scanf("%d", &inputId);

    // Initialize a variable to keep track of whether the ID is found
    int found = 0;

    // Iterate through the studentIds array to find the input ID
    for (int i = 0; i < 10; i++) {
        if (studentIds[i] == inputId) {
            // If the ID is found, output the corresponding score
            printf("Your score is %d.\n", scores[i]);
            found = 1; // Set found to true
            break;     // Exit the loop
        }
    }

    // If the ID is not found, output an error message
    if (!found) {
        printf("The student ID is invalid.\n");
    }

    return 0;
}
```

Q2 [Python] (20 pts): Replace Peaks with Zero

A **"peak"** in an array is an element that is **strictly greater than both of its immediate neighbors**.

Write a function that scans through a list of integers and **replaces every peak with `0`**. The first and last elements of the list are **never considered peaks**, even if they are greater than their only neighbor.

Your function should return the **modified list** with peaks replaced.

Here is an example of the program output:

```
print(replace_peaks([1, 3, 2, 4, 6, 5, 7]))   # Output: [1, 0, 2, 4, 0, 5, 7]
print(replace_peaks([10, 5, 10]))            # Output: [10, 5, 10]
print(replace_peaks([5, 10, 5]))             # Output: [5, 0, 5]
print(replace_peaks([1, 2, 3, 4, 5]))        # Output: [1, 2, 3, 4, 5]
print(replace_peaks([3]))                    # Output: [3]
```

ANS : for loop version

```
def replace_peaks(arr: list[int]) -> list[int]:
    if len(arr) < 3:
        return arr.copy()  # No peaks possible if fewer than 3 elements

    # Make a copy so we don't modify the original during scanning
    result = arr.copy()

    for i in range(1, len(arr) - 1):
        if arr[i] > arr[i - 1] and arr[i] > arr[i + 1]:
            result[i] = 0

    return result

print(replace_peaks([1, 3, 2, 4, 6, 5, 7]))   # Output: [1, 0, 2, 4, 0, 5, 7]
print(replace_peaks([10, 5, 10]))            # Output: [10, 5, 10]
print(replace_peaks([5, 10, 5]))             # Output: [5, 0, 5]
print(replace_peaks([1, 2, 3, 4, 5]))        # Output: [1, 2, 3, 4, 5]
print(replace_peaks([3]))                    # Output: [3]
```

ANS : numpy version

```
import numpy as np

def replace_peaks_numpy_inplace_oneline(arr: list[int]) -> list[int]:
    """
    Replaces peaks in a list of integers with 0 using NumPy, minimizing copies and in one
line for peak detection.
```

```
    A "peak" is an element strictly greater than both of its immediate neighbors.
    The first and last elements are never considered peaks.
    This version minimizes explicit copying and uses a single line for peak detection.

    Args:
      arr: A list of integers.

    Returns:
      A new list with peaks replaced by 0.
    """
    if len(arr) < 3:
      return arr.copy()

    np_arr = np.array(arr) # Convert to NumPy array
    if len(np_arr) < 3: # handle short array again for numpy array input
      return np_arr.tolist()

    # Identify peaks: elements greater than both left and right neighbors in one line
    is_peak = (np_arr[1:-1] > np_arr[0:-2]) & (np_arr[1:-1] > np_arr[2:])

    # Replace peaks with 0 in-place within the NumPy array
    np_arr[1:-1][is_peak] = 0 # Modifying np_arr[1:-1] in-place

    return np_arr.tolist() # Convert the potentially modified NumPy array to a list for return


# Test cases (same as before to verify correctness)
print(replace_peaks_numpy_inplace_oneline([1, 3, 2, 4, 6, 5, 7]))
print(replace_peaks_numpy_inplace_oneline([10, 5, 10]))
print(replace_peaks_numpy_inplace_oneline([5, 10, 5]))
print(replace_peaks_numpy_inplace_oneline([1, 2, 3, 4, 5]))
print(replace_peaks_numpy_inplace_oneline([3]))
```