# CS 101 Autumn 2023 — Quiz 2 — 18 Oct 2023

**5 questions, 48 marks** (Instructor: Suyash P. Awate)

| **Roll Number** | Solutions |
|---|---|
| **Division and Group** | |
| **Name** | |

| Q. No. | Marks | Graded By | Verified By | Student's Cribs |
|---|---|---|---|---|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |
| **TOTAL** | | | | |

**Please read the following instructions carefully before you start.**

- Write your roll number, name, and group number on this page in the space provided. A paper without a roll number and name will NOT be graded.
- Write your answers neatly with a blue/black pen on this question paper itself in the space provided for each question. At the end, you must submit this paper to the invigilator.
- Rough pages will NOT be provided. Use the empty space in the margins.
- Please note that your answers should NOT include any programming concept that hasn't been covered in the class so far. If such answers are found, they shall NOT be graded.
- No clarifications will be provided on any question. When in doubt, make suitable assumptions, state them clearly, and proceed to solve the problem.
- All the best!

1. [8 points]

   Write a declaration for each of the following. (1 point each)

   (a) an array of 8 floats
   ```
   float a[8];
   ```

   (b) an array of 8 pointers to float
   ```
   float* a[8];
   ```

   (c) a pointer to an array of 8 floats
   ```
   float (* a)[8];
   ```

   (d) a pointer to an array of 8 pointers to float
   ```
   float* (* a)[8];
   ```

   (e) a function that returns a float
   ```
   float f();
   ```

   (f) a function that returns a pointer to a float
   ```
   float* f();
   ```

   (g) a pointer to a function that returns a float
   ```
   float (* f)();
   ```

   (h) a pointer to a function that returns a pointer to a float
   ```
   float* (* f)();
   ```

2. [10 points]

What is the output of the following recursive program?

```
#include <iostream>
using namespace std;

void twisted (int n, char first, char middle, char last)
{
    if (n == 1)
    {
        cout << n << first << middle << last << endl;
        return;
    }
    twisted (n - 1, first, middle, last);
    cout << n << first << middle << endl;
    twisted (n - 1, last, middle, first);
    cout << n << middle << last << endl;
    twisted (n - 1, first, middle, last);
}

int main()
{
    twisted (3, 'A', 'B', 'C');
}
```

**Write the Output to the above program here**

Output
1ABC
2AB
1CBA
2BC
1ABC
3AB
1CBA
2CB
1ABC
2BA
1CBA
3BC
1ABC
2AB
1CBA
2BC
1ABC

- 0.6 marks for each correct line and correct order
- the order of the cout statements should be the same as given below
- If an incorrect output is written in between, marks to be given only for the (earlier) outputs i.e. till that line

3. [10 points] The aim of the following program is to generate an array of 10 random numbers, print them on the screen, and then find the maximum value in the array and print it. Each of the following lines may have an error. Indicate which lines have an error (i.e., something that prevents the program from working correctly) and the corrected statement that satisfies the purpose of the program. While doing so: you cannot add any new statements/lines to the program, and you cannot remove any statements/lines from the program, and you cannot change the position of (i.e., move) the lines in the program.

(Each error carries one mark.
Any 10 errors should be pointed out, and corrected code should be written. No marks if corrected code is not written.
No negative marking)

```
01. #include <iostream.h>
02. #include <cstdlib.h>
03. const long int max (const long int, long int) {};
04. main() {
05.     long int a[10] = 0;
06.     srand (2.2);
07.     for (int i = 0; i <= 10; i++) {
08.         a[i] = srand();
09.         cout << a[i] << "\n";
10.     }
11.     long int maximum = RAND_MAX;
12.     for (int i = 0; i < 10; i++);
13.         maximum = max (maximum, a+i);
14.     cout << maximum;
15. }
16. const long int max (const long int a, long int b) {
17.     return a < b ? a : b;
18. }
```

| Line No. | Corrected code |
|---|---|
| 01. | `#include <iostream>` |
| 02. | `#include <cstdlib>` |
| 03. | `const long int max (const long int, long int);` |
| 04. | `int main()` |
| 05. | `long int a[10];` |
| 07. | `for (int i = 0; i < 10; i++)` |
| 08. | `a[i] = rand();` |
| 09. | `std::cout << a[i] << "\n";` |
| 11. | `long int maximum = -1;` |
| 12. | `for(int i = 0; i < 10; i++)` |
| 13. | `maximum = max (maximum, *(a+i));` |
| 14. | `std::cout << maximum;` |
| 17. | `return a > b ?  a :  b;` |

4. [10 points]

Complete the code given below for Selection Sort as taught in the class.
There are 5 blanks. 2 points for each

```
void selectionSort (float data[], int n) {

     // for loop - Blank 1
     for (int i = n-1; i > 0; i--) OR
     for (int i = 0; i <= n-1; i++)
     {
          // code to select an appropriate index
          // of the element in the array 'data'
          // 1 statement only  - Blank 2
          int maxIndex = i;

          // for loop  - Blank 3
          for (int j = i-1; j >= 0; j--) OR
          for (int j = i+1; j < n; j++)
          {
               // code to select an appropriate index
               // of the element in the array 'data'
               // 1 statement only  - Blank 4
               if (data[j] > data[maxIndex]) maxIndex = j; OR
               if (data[j] < data[maxIndex]) maxIndex = j; OR
               if (data[maxIndex] > data[j]) maxIndex = j; OR
               if (data[maxIndex] < data[j]) maxIndex = j;
          }
          /* code to re-arrange the elements in the array 'data' */
          // 1-3 statements only  - Blank 5
          int temp = data[maxIndex];
          data[maxIndex] = data[i];
          data[i] = temp;

     }
}
```

5. [10 points]

Write a recursive function to find the maximum value within an array comprising $N$ integer elements. Assume that $N$ is at least 2. The recursive function must take the array name and the array size as arguments.

```
int max (int arrayName[], int N)
{
     // base case: 1 and only 1 statement
     // Part A: 5 marks

     if (N == 2) return arrayName[0] > arrayName[1] ?
                          arrayName[0] : arrayName[1];

            OR
     if(N==1) return arrayName[0];
     ...

     // recursive step: 1 and only 1 statement
     //Part B: 5 marks

     return max(arrayName[N-1], max(arrayName, N-1));

     OR

     return ( (arrayName[N-1] > max(arrayName, N-1) ) ?
                arrayName[N-1] : max(arrayName, N-1));

}
```

If part A or part B, is not written in 1 statement (ternary operator is not used), then give 2 marks instead of 5 marks.