United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Final Exam:: Trimester: Spring 2023

Course Code: CSE 1111, Course Title: Structured Programming Language

Total Marks: **40** Duration: **2 hours**

[Any examinee found adopting unfair means will be expelled from the trimester/program as per UIU disciplinary rules.]

There are FIVE questions. Answer all the questions. Marks are indicated in the right margin.

Q.1 a) Mr. Tunam is having a very bad semester due to the course named **SPL**. He is very worried about this course because he didn't obtain good marks in the **class tests (CTs)**. Help him to know the marks of his class tests that will be counted in the final grading so that he can prepare adequately for the final exam. Now, **write a C program** based on the following **requirements**: **[4]**

i) Write a function **input_CT_marks(float ct_marks[ ], int n),** where **ct_marks** array stores the marks he has already obtained in all the class tests, n is the number of class tests he has attended, that takes the class test marks from user.

ii) Write a function **take_Highest_CT(float ct_marks[ ], int n)**, which will find and return the highest of the all CT marks.

iii) In the **main()** function, (1) **declare and initialize** the variables and arrays as needed, (2) call the **input_CT_marks** and the **take_Highest_CT** functions, and (3) **display** the returned value.

b) Find the **output** of the following program (left). Notice the **local and global contexts**. **[4]**

```
#include<stdio.h>
int x = 78, y = 0, z = 156;
int first_function(int p, int q) {
    int x = ++p;
    return x + z;
}
void second_function(int w, int y)
{
     y *= x;
     y -= w;
    printf("%d %d %d\n", x, y, z);
    x = first_function(y, z);
    printf("%d %d %d\n", x, y, z);
}
void main() {
    int x = 90;
    y = first_function(x, z);
    printf("%d %d %d\n", x, y, z);
    second_function(y, z);
    printf("%d %d %d\n", x, y, z);
}
```

C Code for 1(b)

```
#include<stdio.h>
#include<string.h>
void main(){
    char str1[100] = "This
    journey is";
    char str2[100] =
    "beautiful";
    int j;
    strncpy(str1, str2, 8);
    for(j=3; str1[j] !='\0';
    j++)
            str1[j] =
    str2[strlen(str1)-j];
    strncat(str2, str1, 3);
    printf("String 1:%s\n",
    str1);
    str2[j-2]='\0';
    printf("String 2: %s\n",
    str2);
    for(j=5; j>2; j--){
            strrev(str1);
    }
    printf("Final: %s", str1);
}
```

C Code for 2(a)

Q.2 a) Find out the **output** of the program right above. **[4]**

b) Suppose, Karim is working as an informer of the Bangladesh Police. He sometimes needs to send important messages to the head office. In order to send a secret message, he has to ensure the security of the message to not make it available to anyone else but the boss. Therefore, instead of sending the message directly, Karim is trying to **change the message** in such a way that no one will be able to read it even if they become successful to hack it. To help Karim, we need to **develop a system** that helps to **change the message**. In the initial stage of building such a system, we want to **replace only the consonants** of the message by the **next consonants** that come in the **alphabet sequence from A to Z**. For example, replace B with C, etc. For Z, replace it with A. **[4]**

Now, **write a C program** that will **change every consonant present in the message to the next one**.

Sample Input	Sample Output
This is Karim reporting. We are ready to start from the zoo.	Uiit it Lasin seqosuioh. Xe ase seaez uo tuasu gson uie aoo.

Q.3 Write a **C program** for a large grocery company to **find the employees** who are eligible for yearly increment. Do the following operations: **[8]**

Structure code

i) **Create a structure** named **Employee** with id(int), name(string), salary(float) and last 12 months' performance scores(int scores[12]) as members.

ii) **Declare** an array of **size N** of type **Employee** structures.

iii) **Take the inputs** from the keyboard.

iv) **Calculate all employees' average performance scores**. For example, if you want to find the average of an employee, you have to add all the 12 scores of one employee and then divide the result by 12. You have to **do it for all the employees**. If the average is **more than 80**, a line will be printed, "Employee with id=(print employee's id here) is eligible for increment." otherwise, print "Employee with id=(print employee's id here) is not eligible for increment."

Q.4 a) Given the following **recursive function** in C, what will be the **output** when the function is called with **num=95**? **[4]**

```
void function(int num) {
    if (num > 0) {
        function(num / 8);
        printf("%d", num % 8);
    }
}
```

b) The dairy milk company **Farm-Fresh** needs your help to compute the **weight and average** daily milk production for each of their cows. The **amount of milk** for each individual cow, **m** is computed as follows: **[4]**

$\mathbf{m = aw^b}$, where, w is calculated as follows:

$\mathbf{w = c(1 - e^{-dx})}$, where x is the age of the cow in years, and a, b, c and d are coefficients.

In your program,

i) **Write a function** that takes the **age (x)**, and the **coefficients (a, b, c, d)** as parameters and compute the **weight and milk production** of a cow. **The function doesn't return anything**. It can take any other parameters as needed.

ii) In the **main** function, **call** the above function and **display** the weight and the average milk production of a cow using the following **data: x=35, a = 0.87, b = 0.45, c = 800, d = 3.5**. Use the concept of **call by reference**.

Q.5 a) **Manually trace** the following code and show the content of array **arr[ ]** in each step: **[4]**

```
#include<stdio.h>
void f1(int* a, int n1, int n2) {
    for (int i = 0; i < n1; i++) {
        if (*(a + i) % 2 == 1) {
            *(a + i + 1) = *(a + i - 1) -(n2 + i);
        }
    }
}
int main() {
    int arr[] = { 6, 3, 2, 7, 0, 1, 5 };
    f1(arr, 7, arr[0] - *(arr + 2));
}
```

b) **Write a C program** that reads integers from a file called "**Sample.txt**", **stores** those values in an array, **calculates** the sum of only those values that are divisible by 7, and finally **writes** it to a file called "**Output.txt**". An example of the contents of Sample.txt and Output.txt is given as follows: **[4]**

Sample.txt

21 -3 -14 7 0 2

Output.txt

Sum: 14