United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Final Exam:: Trimester: Fall 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) Implement a function **updateBalance** that takes **four parameters**: an array of floats representing customer balances, an integer representing the customer's unique identifier (which is the index of the array), an integer representing the type of transaction (1 for withdrawal, 2 for deposit), and a float representing the transaction amount. The function should update the customer's balance based on the **transaction type**, ensuring that withdrawals do not result in a negative balance. **[4]**

i) In the **main** function, **initialize** an array of floats to store the initial balances of **100 customers**. Take the initial balances from user as input.

ii) Take **three values**: an integer (customer identifier), an integer (transaction type), and a float (transaction amount) from user.

iii) Call the **updateBalance** function passing these values. If the transaction is a withdrawal and the withdrawal amount is exceeding the available balance then print "Not sufficient balance" and do not activate the withdrawal.

iv) If transaction is successful in step (iii), print the **updated balance** of the customer.

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

```c
#include <stdio.h>
int ara[5], x = 20;
void change(int p) {
   --p;
   p--;
}
void update(int n) {
   for(int i = n - 1; i >= 0; i--){
      ara[i] -= x;
      change(x);
   }
}
void main() {
    int n = 5;
    for(int i = 0; i < n; i++){
      ara[i] = (i + 5) * 2;
    }
    update(n);
    for(int i = 0; i < n; i++){
      printf("%d, ", ara[i]);
    }
}
```

C Code for **1(b)**

```c
#include <stdio.h>
#include <string.h>
int main(){
   char str1[50]="CSE-1111 SPL";
   char str2[50]="I am a UIUian";

   int i=strlen(str1) * 0.5 - 2;

   for(int m=0; i+m<strlen(str1); m+=3){
         str1[i+m]=str2[m];
   }

   strcat(str1, str2);

   if(strcmp(str2, str1)>0){
     strncat(str1, "CSE is awesome.",6);
   }
   else{
     strncat(str2, "CSE is awesome.",6);
   }
   return 0;
}
```

C Code for **2(a)**

Q.2 a) Show **manual tracing** (every change) of variables i, m, str1, and str2 of the program above at right. **[4]**

b) Write a **C program** that takes a string from the keyboard. The program will **count (case insensitively)** the number of different vowels and **display** the statistics as shown below. **[4]**

Sample Input	Sample Output
Owls fly high above the clouds.	A/a: 1 E/e: 2 I/i: 1 O/o: 3 U/u: 1

Q.3 Suppose you're developing a program for the **United International University (UIU) Bookshop** to manage customer purchases. Your task is to create a **C program** that can store, manage and manipulate the bookshop's **last 12 months' customer data**. As it is the end of the year, the bookshop is giving a "**best customer award**" to the customers who purchased the most in the past year. Write a C program that will: **[8]**

1. **Store** the following information of a customer in a structure.
   (i) Name (ii) ID (iii) Number of times shopped, and (iv) An array of spent money in each of the shoppings. Use **appropriate data types and variable names** for all the features.
2. **Take input** for **100 customers** from the user.
3. **Calculate** the **average purchase for each customer** (total money spent divided by the total number of shopping).
4. To **find the best customer**, only consider the customers who have shopped **more than 10 times**. Among these selected customers, the customer having the **best average purchase (calculated from (3))** will win the award. **Print** the **customer's name** who has won the award.

Q.4 a) Write a **C program** that does the following: **[4]**

- Declare an integer variable **num** and initialize it with any value.
- Declare a **pointer variable** and **assign** the address of **num** to it.
- Use the **pointer** to double the value of **num**.
- **Print** the value of **num** both before and after the modification.

b) Find the **output** of the code provided below on the left.

```
#include <stdio.h>
void modifyArray(int arr[], int
size){
   for (int i = 0; i < size; i++){
        arr[i] *= 2;
   }
}
void main() {
   int numbers[] = {1, 2, 3, 4, 5};
   int *ptr = numbers;
   modifyArray(ptr, 5);
   for(int i = 0; i < 5; i++){
      *(ptr+i) = *(ptr+i) + i;
      printf("%d ", *(ptr+i));
   }
}
```

C Code for **4(b)**

```
#include <stdio.h>
void go(int num){
    printf("How are you?\n");
    if(num==0){
        return;
    }
    num /= 10;
    go(num);
    printf("I am fine\n");
}

int main(void){
    int num = 45633;
    go(num);
    return 0;
}
```

C Code for **5(a)**

Q.5 a) Write the **output** of the program provided above on the right. **[4]**

b) Write a **C program** that performs the following tasks: **[4]**

1) **Read** the following "**in.txt**" file that has integer numbers on separate lines. **Find** the maximum value of those numbers.
2) **Create** a new file "**out.txt**" and **print** the maximum number into the new file.
3) Remember to **display appropriate messages** if the input or the output file couldn't be opened.

in.txt
1 234 3 56 .. .. 61 10