PDS Lab

Assignment 8, Structures and Pointer to Structures

October 29, 2024

Total Marks: 100

Instructions

- Create a directory named as Lab-8.
- Give the name of the program as $< p.c >$ where $< p >$ implies the problem number, like 1.c, 2.c, 3.c, etc. Store all the programs of this week under this directory.
- You should upload all .c files (1.c, 2.c, 3.c) as a single Zip file to the Moodle course web page latest by 4.45 PM (without penalty). The cutoff time will be till **5.00 PM** with a penalty of 25% on your secured marks (i.e., if you secured 80 marks, after penalty you will get 60 marks). Beyond 5.00 PM, the moodle system will not allow you to submit, as a result you will get zero.

---

1. **Leader Selection Problem**. In a galaxy far, far way there is an alien community where the leader is selected based on the age. Each person in the community has a name (array of characters without any space) and an age split into an array of three complex numbers. The final age is the average of the modulus of these three complex numbers. Leader must have the highest final age. If two or more persons have same age, one is selected at random. (40)

   Given a population size input from the user, write a program that outputs the name of the leader.

   - Modulus of a complex number: The square root of the sum of the squares of the real part and the imaginary part of the complex number i.e., $\sqrt{(x^2 + y^2)}$
   - You can assume that the name is upto 50 chars.
   - You can assume both real and imaginary parts of a complex number to be `float`.
   - You can assume that age can be of float type in the community.

   Use the following starter code for your program.

```c
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

float modulus(Complex c)
{
   // Complete this code
}

int main()
{
 int population_size, num_of_complex_nums_in_age = 3;
 printf("Enter the population size of aliens: ");
 scanf("%d", &population_size);

 Alien *aliens = (Alien *)malloc(population_size * sizeof(Alien));
 // Write rest of the code
}
```

The following shows two sample runs:

Run 1

```
Enter the population size of aliens: 3
Enter name for alien 1: Nien
        Enter complex number (real, imaginary) in age[0]: 2.1 3.2
        Enter complex number (real, imaginary) in age[1]: 2.3 4.5
        Enter complex number (real, imaginary) in age[2]: 2.4 6.7
Enter name for alien 2: Sullustan
        Enter complex number (real, imaginary) in age[0]: 4.1 3.2
        Enter complex number (real, imaginary) in age[1]: 2.3 45.6
        Enter complex number (real, imaginary) in age[2]: 2.44 63.7
Enter name for alien 3: Grakkus
        Enter complex number (real, imaginary) in age[0]: 2.4 6.7
        Enter complex number (real, imaginary) in age[1]: 4.5 2.3
        Enter complex number (real, imaginary) in age[2]: 2.1 3.2

Finding for 'Nien':
        Age --> 5.332708
Finding for 'Sullustan':
        Age --> 38.201881
Finding for 'Grakkus':
        Age --> 5.332708

Selected 'Sullustan' as leader
```

Run 2

```
Enter the population size of aliens: 3
```

```
Enter name for alien 1: Nien
        Enter complex number (real, imaginary) in age[0]: 2.1 3.2
        Enter complex number (real, imaginary) in age[1]: 2.3 4.5
        Enter complex number (real, imaginary) in age[2]: 2.4 6.7
Enter name for alien 2: Zingo
        Enter complex number (real, imaginary) in age[0]: 6.1 7.9
        Enter complex number (real, imaginary) in age[1]: 1.3 4.6
        Enter complex number (real, imaginary) in age[2]: -2.4 1.7
Enter name for alien 3: Bingo
        Enter complex number (real, imaginary) in age[0]: 6.1 7.9
        Enter complex number (real, imaginary) in age[1]: 4.6 1.3
        Enter complex number (real, imaginary) in age[2]: 1.7 -2.4

Finding for Nien:
        Age --> 5.332708
Finding for Zingo:
        Age --> 5.900746
Finding for Bingo:
        Age --> 5.900746

The following  2 persons have same age
Zingo
Bingo

Randomly selected 'Bingo' as leader
```

2. A sparse matrix is a matrix with a large number of zero entries. Write a program that takes two sparse matrices as input from the user and finds their sum. (40)

   - You **must** represent a matrix using an array of structures, where each structure contains a row number, column number, and value.
   - Ask the user about the number of non-zero elements in the matrices and only input those values.
   - Write a function with pointer to the matrices as parameters (additional parameters are allowed as long as you access the matrices only through the parameters) that finds their sum and returns the result.
   - Print the input matrices and their sum.

   You can assume that the maximum number of elements in the matrices is 100.

   Example:

$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 3 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

```c
#include <stdio.h>
#include <stdlib.h>
#define MAX_ELEMENTS = 100

int main() {
 int rows, cols;
 SparseMatrix m1, m2, result;

 printf("Enter the number of rows and columns for the matrices: ");
 scanf("%d %d", &rows, &cols);

 // Ask the user about the number of non-zero elements in each matrix.
 // Input those values in the matrix

 // Call the function to add m1, m2 and put it in result

 // Print all three matrices
}
```

For reference, the following is an example run:

```
Enter the number of rows and columns for the matrices: 3 3
Enter the number of non-zero elements in First matrix: 3
Enter the number of non-zero elements in Second matrix: 2

---- Taking input for the first matrix -----
Enter 3 non-zero elements
        Enter row, column, and value: 0 0 2
        Enter row, column, and value: 1 1 1
        Enter row, column, and value: 2 2 3

---- Taking input for the second matrix -----
Enter 2 non-zero elements
        Enter row, column, and value: 0 0 1
        Enter row, column, and value: 0 2 2
Adding Matrix
        2    0    0
        0    1    0
        0    0    3
and
        1    0    2
        0    0    0
        0    0    0
results in
        3    0    2
        0    1    0
        0    0    3
```

3. Given two integer numbers of arbitrary length, perform the operations: +, -, *. (20)
Example that shows the + operation.

```
Input two numbers:
121212122344221221
212121332311122212
Output:
333333454655343433
```