Lab # 1

Object Oriented Programming

BS SE F17 Morning

Instructions:

- Indent your code properly.
- Use meaningful variable and function names. Follow the naming conventions.
- Use meaningful prompt lines and labels for all input/output.
- Make sure that there are **NO dangling pointers** or **memory leaks** in your program.

Task 1 **5 marks**

Write a C++ function which takes an array of integers and its size as arguments. Firstly, the function should determine the number of positive values present in the array. After that the function should dynamically allocate an array of doubles to store those many values.

Then, the function will copy all the positive values from the original array into the newly allocated array. At the end, the function should return a pointer to the dynamically allocated array containing all the positive values.

The size of this new array will be returned through a reference parameter (see the 3rd argument in the function prototype given below).

The prototype of your function should be:

int* extractPositives(int* original, int origSize, int& newSize);

Task 2 (Minesweeper) **10 marks**

Have you ever played Minesweeper? It's a cute little game which within a certain Operating Systems which name we can't really remember. Well, the goal of the game is to find where are all the mines within a M*N field. To help you, the game shows a number in a square which tells you how many mines there are adjacent to that square. For instance, suppose the following 4*4 field with 2 mines (which are represented by an * character).

***** . . .
. . . .
. ***** . .
. . . .

If we would represent the same field placing the hint numbers described above, we would end up with:

* 1 0 0
2 2 1 0
1 * 1 0
1 1 1 0

As you may have already noticed, each cell may have at most 8 adjacent cells.

The Input:

The first line of the input will contains the number of rows and columns respectively. The next n lines contain exactly m characters and represent the field. Each safe square is represented by an ".” Character (without quotes) and each mine square is represented by an "*" character (also without the quotes).

The Output:

The output should contain the field with the ".” Character replaced by the number of adjacent mines to that square.

Sample Input:

4 4

***** . . .
. . . .
. ***** . .
. . . .

Sample Output:

* 1 0 0
2 2 1 0
1 * 1 0
1 1 1 0

Task 3

Weather Statistics **5 marks**

Write a program that uses a structure to store the following weather data for a particular month:

Total Rainfall
High Temperature
Low Temperature
Average Temperature

The program should have an array of 12 structures to hold weather data for an entire year. When the program runs, it should ask the user to enter data for each month. (The average temperature should be calculated.) Once the data are entered for all the months, the program should calculate and display the average monthly rainfall, the total rainfall for the year, the highest and lowest temperatures for the year (and the months they occurred in), and the average of all the monthly average temperatures.

Input Validation: Only accept temperatures within the range between –100 and +140 degrees Fahrenheit.

Task 4 **5 marks**

Write a program that declares a struct to store the data of a football player **(player's name, player's position, number of touchdowns, number of catches, number of passing yards, number of receiving yards, and the number of rushing yards)**. Declare an array of 10 components to store the data of 10 football players. Your program must contain a function to input data and a function to output data. Add functions to search the array to find the index of a specific player, and update the data of a player. Your program should be menu driven, giving the user various choices.