Assignment #5 (40 points; due Tuesday, 11/7/2023, at 11:59:00 P.M.)

For all Programs:

To get in the habit of writing pseudocode write the simple pseudocode for these programs. Put your pseudocode in comments at the top of your programs, as well as many lines within the code necessary to describe what's going on in the program. PSEUDOCODE IS REQUIRED FOR ALL PROGRAMS.

For each program make sure and include the following comments at the top (do this on all homework assignments from now on – this is required):

```
//Your Name
//CS 1, Section #ABC
//Assignment #X, Problem #Y
//Summary of the program
```

For this homework you have two programs so you will be emailing 3 program files. These programs address Chapters 6, 7, and 9.

Lottery Checker

1. (16 points) You're going to write a program that models the Littleton City Lotto (not a real Lotto game). The program is going to allow to user to first select their lotto numbers. The program will then randomly generate the winning lotto numbers for the week and then check the winning numbers against the random ticket the user played in the Lotto to see how many numbers the user guessed correctly.

The rules for lotto work as follows:

1) Select 7 numbers between 1 and 40
2) Twice every week 7 numbers are drawn at random
3) If a player matches all 7 numbers they win a million dollar prize
4) If a player matches 6 numbers they win $100,000
5) If a player matches 5 numbers they win $5,000
6) If a player matches 4 numbers they win $100.
7) If a player matches 3 numbers they win a free ticket.

Your program should work as follows. (**NOTE:** I have listed some functions you need to include in your program – you may include other functions if you would like and they make your program more readable/efficient).

Step 1

Create an array named UserTicket to hold each of the user's lotto number selections.
Create an array named WinningNums to hold the winning lotto numbers.

Step 2

Display the following menu:

LITTLETON CITY LOTTO MODEL:
----
1) Play Lotto
q) Quit Program
Please make a selection:

If the selection is 1:

a. First ask the user their name and store it in an appropriate variable.
b. Next, call a function named getLottoPicks that asks the user to enter their 7 lotto number picks (selections) for the week. Each of the user's lotto picks should be stored in the UserTicket array. The lotto does NOT have duplicate numbers in it. Find a way to not allow duplicate numbers to be picked by the user. You may want to create another function called NoDuplicates that checks to see if the user's selection is already in the UserTicket array. If the user enters a number already in the array, ask them to enter another number until they enter one that is not a duplicate. This means the UserTicket array should contain no duplicate numbers.
c. Next, call a function named GenWinNums that randomly generates the winning lotto numbers for the week based on the rules stated above and stores the winning lotto numbers in the WinningNums array (so you are going to fill the WinningNums array with random numbers between 1 and 40). Do not allow this function to generate duplicate winning numbers (if you design your NoDuplicates function above well you should be able to re-use it to check for duplicates in the WinningNums array). **HINT:** There are two ways to avoid having duplicate numbers in an array. Most of you will need to use the first method. The first method is to check the array after each number is generated or entered to make sure that number is not already in the array. If the number generated/entered is already in the array a new number should be generated/entered. The second method involves sorting and checking that the numbers next to each other in the array after the sort are not the same – if you have taken CSIS 130 and know how to sort an array you can use this method.

d. The next step is to check the user's lotto ticket (represented by the UserTicket array) to see if they have won any prizes in the Lotto game. Check each number the UserTicket array to see if that number is in the WinningNums array and count how many numbers are matched.

Display a report similar to the following showing user's lotto results – the example output below assumes the user entered a name of "Julie" when the program started.

In the "Winnings" section of the report
Display: JACKPOT!!! - $1 MILLION if all 7 numbers were correct
Display: GREAT! - $100,000 if 6 numbers were correct
Display: LUCKY YOU! - $5,000 if 5 numbers were correct
Display: NOT BAD - $100 if 4 numbers were correct
Display: FREE TICKET if 3 numbers were correct
Display: SORRY NOTHING if 2 or less numbers were correct

JULIE'S LOTTO RESULTS

WINNING TICKET NUMBERS: 35 03 01 15 10 25 22
JULIE'S TICKET: 33 15 02 06 21 20 19

RESULTS:

Number Matches: 1
Winnings : SORRY NOTHING

If the selection is q: Quit the program
If the selection is not q and not 1: Display an invalid selection message

Allow the user to play lotto as many times as they would like.

Movie Theater Tickets

2. (16 points) Write a program that can be used by a small theater to sell tickets for performances. The theater's auditorium has 15 rows of seats with 20 seats in each row.

Step 1

The program should have a FUNCTION showSeats that displays a screen that shows which seats are available and which are taken. Seats that are taken should be represented by a # symbol and seats that are available should be represented by a * symbol. The first thing your program should do is initialize all of the seats to available (*) and display the seating chart. (HINT: The seating chart should be a two dimensional array.) Here is an example of the seating chart with all seats initialized to available.

```
Seats:   0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17 18 19
Row 0    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 1    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 2    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 3    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 4    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 5    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 6    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 7    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 8    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 9    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 10   *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 11   *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 12   *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 13   *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
Row 14   *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
```

Step 2

Each row in the auditorium has a different ticket price. So tickets in row 0 may be 5.00 each and tickets in row 1 may be 10.00 each. Your program should have a FUNCTION readPrices that reads the ticket price of each row from an input file called prices.txt. The ticket price for each row should be stored in a one dimensional array.

Step 3

Your program should have variables tracking the total number of tickets sold and the total revenue for all tickets sold.

Step 4

Your program should allow the user to sell tickets one at a time. The user should be able to sell as many tickets as they would like (you need a loop for this). Do this with

some sort of prompt or menu asking the user if they would like to sell another ticket. Don't forget to validate input data if you need to.

To allow the user to sell a ticket your program should have the user enter a row number and a seat number for the ticket they would like to sell. The program should do four things with this information:

1. It should check to see if the seat is available. If the seat is taken the program should not allow the user to sell the ticket. If this happens, print a message to the user saying the ticket is not available and prompt the user to see if they would like to sell another ticket.

2. If the seat is available the program should update the seating chart by putting a taken symbol (#) in that seat's position in the chart.

3. The program should then look up the row price for the seat sold. Your program should have a variable tracking the total revenue, the price of the seat sold should be added to this total after each sale.

4. Your program should have a variable tracking the total tickets sold. The next thing your program should do when selling a ticket is update the total tickets sold.

Step 5

Once the user is finished selling tickets print out an updated seating chart followed by the total tickets sold and the total revenue generate from those tickets.

NOTE: You are required to use two arrays in this program, one for the seating chart and one to store the prices for each row. You are also required to use two functions: one to display the seating chart and one to read in the price per row data and store it in the array with the prices for each row in it. You may use other functions if you want to but they are not required.

Median Function

3. (8 points) In statistics, when a set of values is sorted in ascending or descending order, its median is the middle value. If the set contains an even number of values, the median is the mean, or average, of the two middle values. Write a function that accepts as arguments the following:

    a. An array of integers

    b. An integer that indicates the number of elements in the array

The function should determine the median of the array. This value should be returned as a double. (Assume the values in the array are already sorted.) You'll create two arrays: one with three elements (e.g. 1, 3, 7) and one with four elements (e.g. 2, 4, 5, 9). Demonstrate your pointer prowess by using pointer notation instead of array notation in this function. (In other words, you cannot access an array at any point using array[1], for example. It must be *(array + 1).)