CSC120
Homework 11
Due Thursday, 11/30/2017

This homework consists of three parts. For each part submit a WELL COMMENTED, INDENTED PROGRAM. Work Independently.

Part 1 Arrays

Write a program that simulates a lottery. Your program should have an array of five integers named lotteryNumbers. Use the Random class to generate a random number in the range of 0 through 9 for each element in that array. Your program should have another array with five integers called user. Write a loop to read 5 integers in from the keyboard to put in user array.

The program should then display the count of digits that match between the user array and the lotteryNumbers array.

Here is a sample of the output

Welcome to play Lottery ...
enter your lottery number
3
enter your lottery number
4
enter your lottery number
5
enter your lottery number
6
enter your lottery number
7

The Lottery Number is:
2 3 8 3 9
Your Number is:
3 4 5 6 7
There are 1 matching digits.

Hints:
1. Define 2 arrays -
   int [] user = new int [5];
   int [] lottery = new int [5];
2. Declare a variable to count the number of matches
3. Write out prompt to enter numbers
4. Read in numbers in a loop into user array
5. Generate random numbers in a loop into lottery array lottery
6. Print out the contents of both arrays
   example for lottery - include code to print BOTH arrays out
   for(int i=0;i<lottery.length;i++)
       System.out.print(lottery[i] + " ");
7. Use nested loops to take every number in lottery to compare to every number in user. If you find a match do these three things:
   a. increment the counter
   b. change the number in the user array so you don’t count it again
   c. break so you don’t look any further
8. Print out number of matches

Part 2 ArrayPractice

This program will require you to:
1. Ask user to enter the length of the array
2. Read in that length and use it to create an array of integers
3. Write a loop that asks the user to enter an integer and then reads it into the array
4. Write a loop that finds the max of the array
5. Write a loop to find the min of the array
6. Write a loop to find the average of the array (be careful with int division)
7. Write a loop to find the number of numbers in array divisible by 5
8. Write a loop to print the array
9. Write a loop to print the array backwards
10. Find the smallest two numbers in the array

Example output

----jGRASP exec: java Loop
Enter array length
7
Enter value
21
Enter value
23
Enter value
56
Enter value
76
Enter value
89
Enter value
45
Enter value
34
The max is 89  
The min is 21  
The average is 49.142857142857146  
The number of numbers divisible by 5 is  is 1  
The array forward is 21, 23, 56, 76, 89, 45, 34,  
The array reversed is 34, 45, 89, 76, 56, 23, 21,  
The smallest two numbers are 21 and 23

You can get 1 extra credit point if you figure out how to not have the last number be followed by a ,

Part 3 two-dimensional Array

1) Create a program that uses an array to calculate distance between cities based on the chart below. You should ask the user to enter a starting point city then enter a destination city. You can have the user enter numbers instead of typing city names for example, Enter 1 for Boston, 2 for Chicago, 3 for Dallas, etc. These numbers become the indexes into your two-dimensional array

(Hint: This requires a 10 by 10 integer array (distance array). Do not include the city names in your array.
Here is a table that you can use to initialize the distance array)

<table>
<thead>
<tr>
<th></th>
<th>Boston</th>
<th>Chicago</th>
<th>Dallas</th>
<th>Reno</th>
<th>Los Angeles</th>
<th>Miami</th>
<th>New Orleans</th>
<th>Toronto</th>
<th>Vancouver</th>
<th>Wash DC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boston</td>
<td>0</td>
<td>1004</td>
<td>1753</td>
<td>2752</td>
<td>3017</td>
<td>1520</td>
<td>1507</td>
<td>609</td>
<td>3155</td>
<td>448</td>
</tr>
<tr>
<td>Chicago</td>
<td>1004</td>
<td>0</td>
<td>921</td>
<td>1780</td>
<td>2048</td>
<td>1397</td>
<td>919</td>
<td>515</td>
<td>2176</td>
<td>709</td>
</tr>
<tr>
<td>Dallas</td>
<td>1753</td>
<td>921</td>
<td>0</td>
<td>1230</td>
<td>1399</td>
<td>1343</td>
<td>517</td>
<td>1435</td>
<td>2234</td>
<td>1307</td>
</tr>
<tr>
<td>Reno</td>
<td>2752</td>
<td>1780</td>
<td>1230</td>
<td>0</td>
<td>272</td>
<td>2570</td>
<td>1732</td>
<td>2251</td>
<td>1322</td>
<td>2420</td>
</tr>
<tr>
<td>Los Angeles</td>
<td>3017</td>
<td>2048</td>
<td>1399</td>
<td>272</td>
<td>0</td>
<td>2716</td>
<td>1858</td>
<td>2523</td>
<td>1278</td>
<td>2646</td>
</tr>
<tr>
<td>Miami</td>
<td>1520</td>
<td>1397</td>
<td>1343</td>
<td>2570</td>
<td>2716</td>
<td>0</td>
<td>860</td>
<td>1494</td>
<td>3447</td>
<td>1057</td>
</tr>
</tbody>
</table>
Here is an example of the output

```
---jGRASP exec: java CityDist

Possible city names:
Enter 0 for Boston
Enter 1 for Chicago
Enter 2 for Dallas
Enter 3 for Reno
Enter 4 for Los Angeles
Enter 5 for Miami
Enter 6 for New Orleans
Enter 7 for Toronto
Enter 8 for Vancouver
Enter 9 for Washington DC
```

Enter a starting point city:
0
Enter a destination city:
7

The distance between the cities is 609 miles