Lab 13  Name:_________________________ Checked:_____

Objectives:
Practice using two dimensional arrays to store and process values of different types.

a) A Simple 2D array example
//********************************************************************
// TwoDArray.java       Author: Lewis/Loftus
// Demonstrates the use of a two-dimensional array.
//********************************************************************
public class TwoDArray
{
    // Creates a 2D array of integers, fills it with increasing
    // integer values, then prints them out.
    public static void main (String[] args)
    {
        int[][] table = new int[5][10];
        // Load the table with values
        for (int row=0; row < table.length; row++)
            for (int col=0; col < table[row].length; col++)
                table[row][col] = row * 10 + col;
        // Print the table
        for (int row=0; row < table.length; row++)
        {
            for (int col=0; col < table[row].length; col++)
                System.out.print (table[row][col] + "	");
            System.out.println();
        }
    }
}

• Run this program and observe what it does.

• The output produced is shown below. Circle the entries for table[0][5] and table[3][2]

0  1  2  3  4  5  6  7  8  9
10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29
30 31 32 33 34 35 36 37 38 39
40 41 42 43 44 45 46 47 48 49

• If we were to draw a picture of the array, labeling the rows and columns with the array indices, it would look like this:

<table>
<thead>
<tr>
<th>#</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
</tr>
<tr>
<td>4</td>
<td>40</td>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
</tr>
</tbody>
</table>
• Rename the program Lab13a.java and modify it so that its output actually looks EXACTLY as shown above (with labels above and to the left).

• Modify the dimensions of the array – make it 3 rows by 4 columns and run the program again. The output should look right without having to change anything else in the program. If necessary, adapt your program so that it works with any reasonable dimensions (note that there is a limit to how many columns can be displayed across on one line, so it is not expected to work well with large values for the number of columns).

• What happens if you do not initialize the array’s values?

Answer: ________________________________________________________________

b) A 2D array of double
Make a new version of your program from part (a) and name it Lab13b.java that creates instead a 2D array of 5x5 values of type double, set to random values in the range 0….1 (use Math.random()).

• What happens if you do not initialize the array’s values?

Answer: ________________________________________________________________

c) A 2D array of boolean
Make a new version of your program from part (a) and name it Lab13c.java that creates instead a 2D array of 5x5 values of type boolean.

1) Suppose indices represent people and that the value at row i, column j of a 2D array is true just in case i and j are friends and false otherwise. Use initializer list to instantiate and initialize your array to represent the following configuration:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>1</td>
<td>*</td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

(* means “friends”)

2) Write some code to count how many pairs of friends are represented in the array. Note that each friendship pair appears twice in the array, so in the example above there are 6 pairs of friends).

3) Write a method to check whether two people have a common friend. For example, in the example above, 0 and 4 are both friends with 3 (so they have a common friend), whereas 1 and 2 have no common friends. The method should have three parameters: a 2D array of boolean representing the friendship relationships and two integers i, j. The method should return true if there is an integer k such that i is a friend of k and k is a friend of j and return false otherwise. Make this method static and test it from the main() method of your program Lab13c.java.