Mr. Bellavia

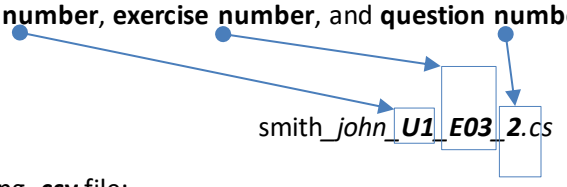
EXERCISE 03 – 2D ARRAYS

IMPORTANT: Before submission, make a copy of your **'Program.cs'** file for each question and then rename each file to the following:

File Names:

- *last_name_first name*_U4_E03_1.cs
- *last_name_first name*_U4_E03_2.cs
- *last_name_first name*_U4_E03_3.cs

Note: Along with last name and first name, make sure the end of the filename (i.e., before the **.cs**) has the **unit number**, **exercise number**, and **question number.** For example:



1. Consider the following **.csv** file:

input.csv:

```
ID,Last Name,First Name,Phone Number,Minutes
1,Doe,John,905-555-5555,878
2,Barnett,Courtney,905-666-6666,112
3,Morrison,Jim,905-777-7777,912
4,Doe,Jane,905-222-2222,1020
5,Mitchell,Joni,416-333-3333,340
```

The **first row** are the **column names,** and the **next 5 rows** are the **data.** Read in the **data** from this **.csv** file into a **5 x 5 string array** using a **nested for-loop.** Create **another nested for-loop** to **output** all data in a table format. Also, during your second nested for-loop, calculate the minimum, maximum, and average minutes from the data and then output to the screen. Your output should look like the following:

Sample **Output:**

```
ID Last Name  First Name Phone Number    Minutes
-------------------------------------------------------------
1  Doe        John       905-555-5555    878
2  Barnett    Courtney   905-666-6666    112
3  Morrison   Jim        905-777-7777    912
4  Doe        Jane       905-222-2222    1020
5  Mitchell   Joni       416-333-3333    340
-------------------------------------------------------------
Minimum: Courtney Barnett 112
Maximum: Jane Doe 1020
Average: 652.40
```

Note: Don't worry if the columns do not line up perfectly.

2. Consider the following .csv file:

input.csv:

```
ID,Last Name,First Name,Phone Number,Minutes
1,Doe,John,905-555-5555,1020
2,Barnett,Courtney,905-666-6666,112
3,Morrison,Jim,905-777-7777,912
4,Doe,Jane,905-222-2222,1020
5,Mitchell,Joni,416-333-3333,112
```

Notice now that there are minimums and maximums that are the same (i.e., **112** and **1020**). Re-write your program from question 1 so that your program can output multiple minimum and maximums from any given set of data.

Sample **Output:**

```
ID Last Name  First Name Phone Number    Minutes
------------------------------------------------------------
1  Doe        John       905-555-5555    1020
2  Barnett    Courtney   905-666-6666    112
3  Morrison   Jim        905-777-7777    912
4  Doe        Jane       905-222-2222    1020
5  Mitchell   Joni       416-333-3333    112
------------------------------------------------------------
Minimum: Courtney Barnett, Joni Mitchell 112
Maximum: John Doe, Jane Doe 1020
Average: 652.40
```

Mr. Bellavia

3. 2D Array Mapping Exercise:

Do the following steps in order…

a) Write a program that creates a 5 x 5 array to hold a set of letters (chars).

b) Traverse this array and store random letters in each cell of this array by using the following code:

```
Random rnd = new Random();
char randomChar = (char)rnd.Next('a','z');
```

c) Make sure the program does not store the same letter twice while traversing. **Hint:** You will need to use **two nested for-loops** to accomplish this. For example:

```
for(int x = 0; x < 5; x++)
{
   for(int y = 0; y < 5; y++)
   {
      //generate a new char…
      //then check if this char already exists in your array
      //  with another nested for-loop:
      for(…
      {
         for(…
         {
            …
         }
      }
      //if the char already exists in the array then subtract 1
      //  from 'y' so that the current iteration of the outer
      //  nested for-loop repeats
   }
}
```

d) Output this 2d-array to the screen (another nested for-loop).

e) Now have the program **repeatedly** (i.e., using a while-loop) ask the user to type in a coordinate (x, y) followed by a letter to be placed at that coordinate.
   - Your program should replace the existing letter at that coordinate, but only if that letter does not exist anywhere else in the array. If that letter already exists in the array, then tell the user.
   - Output the new array to the screen.
   - Repeat this process until the user types a '*-1*' for 'x' or 'y' in which case your program should end.