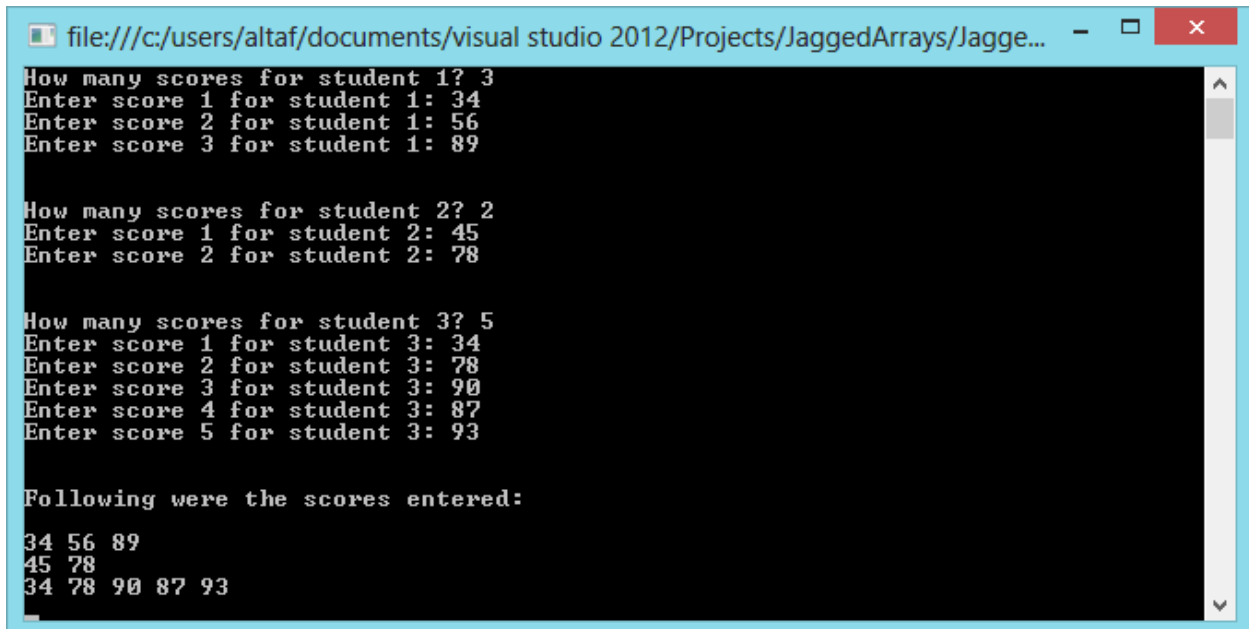


CIS 345 – Business Information Systems Development II – Fall 2014

In-class Exercise 5: Jagged Arrays

Due on Blackboard: Today, Monday, September 29 by 10:00 PM

Design a C# program that accepts a variable number of test scores for a set of 3 students. The program should ask the user how many scores he or she would like to enter for each student. Use Jagged Arrays to implement the varying number of scores per student.



```
file:///c:/users/altaf/documents/visual studio 2012/Projects/JaggedArrays/Jagge...  
How many scores for student 1? 3  
Enter score 1 for student 1: 34  
Enter score 2 for student 1: 56  
Enter score 3 for student 1: 89  
  
How many scores for student 2? 2  
Enter score 1 for student 2: 45  
Enter score 2 for student 2: 78  
  
How many scores for student 3? 5  
Enter score 1 for student 3: 34  
Enter score 2 for student 3: 78  
Enter score 3 for student 3: 90  
Enter score 4 for student 3: 87  
Enter score 5 for student 3: 93  
  
Following were the scores entered:  
34 56 89  
45 78  
34 78 90 87 93
```

Create a project called JaggedArrayApp and rename Program.cs to JaggedArrays.cs.

Implement the following methods within your class. Methods will be static.

- ***Method Name:*** ReadScores

Purpose: ReadScores prompts the user to enter scores and stores them into an array. ReadScores expects to be given as parameters, a student number, and an integer array to put the scores into. It will use the student number in its prompt to the user and will then put scores into the array provided to it.

Access Modifier: Private

Parameters: an integer called “studentNumber” and an array of integers called “scoreArray”

Return value: None

General method logic:

Loop through the scoreArray (*use the Length property of the array in the condition*)

Prompt the user to enter score *x* for student *y*. *Use the student number parameter and the counter variable for the score #. Use placeholders!*

Read an integer. Assign it to the current element of the array. *Use the counter variable as the index.*

- ***Method Name:*** PrintAllScores

Purpose: This method prints to the console all the values found in a jagged array. It expects the jagged array to be provided to it as a parameter.

Access Modifier: Private

Parameters: a jagged array of integers called scoresArray

Return value: None

General method logic:

Tell the user that you are going to show the scores found in the jagged array.

Loop through scoresArray (*use a row counter, i. Works just like a single dimensional array!*)

Loop through each array within each element of scoresArray

Use j as counter. scoresArray[i] is an array! It has a length. So, use arrayName[counter].Length as part of the condition

Print the value inside the inner array. Access it by referring to arrayName[i][j]

Print a new line!

- *Method Name:* Main

Purpose: This method will ask the user how many scores they would like entered for each student. It will then call other methods to read the scores and print out the scores back to the user.

General method logic:

Declare a jagged array of integers, called “jaggedArray” and instantiate it with three rows. *This mean you are only specifying the number of rows and leaving the number of columns blank!*

Declare an integer, scoreCount and initialize it to zero.

Loop through the jaggedArray. *Use the Length property.*

Prompt the user for the number of scores for Student x.

Read the number of scores and store it in scoreCount.

Instantiate a new array of integers. Use the scoreCount variable as the number of elements for this array. Assign the reference to this new array to the element i of jaggedArray. *ArrayName[counter] is the structure to access an element of an array!*

Call ReadScores. Supply it with the student number (use the counter + 1) and the current element (i) of the jaggedArray as arguments.

Print new lines as needed per row.

Call the PrintAllScores method. Pass it your jagged array as an argument.

Call ReadLine to insert a pause.

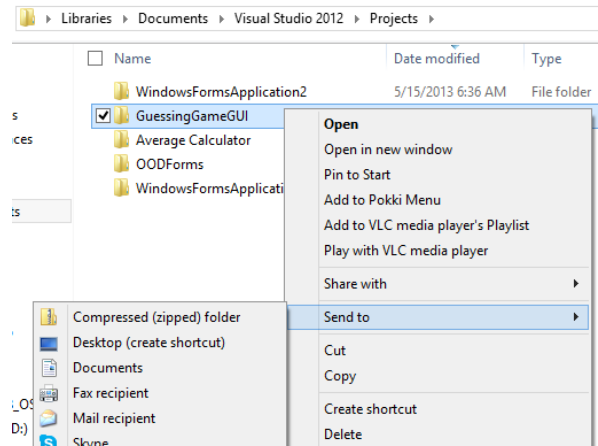
Optional Enhancements and Challenges

- Write a method that calculates the average of all scores. *This is not overly complex since you can automate how many times you have to loop by using the Length property. (This can be integrated right into the PrintAllScores method!)*
- Write a method that calculates the average of score x (ask user for which score). *This is slightly more complex since score x might not exist for all students, e.g. some students do have a Score 5 but some have only 3 scores, so you cannot factor those other students in, when calculating score 5.*
- Don't ask how many scores the user would like to enter for a student. *Let the user surprise you!*
Accept the list of scores as a space-separated list of values e.g. "100 94 78 89" or "78 23". Split those scores using a space character as a delimiter by using the Split method of the String class. After this point, assess how many elements there are in the resulting array and based on that, you should create the "columns" array (in the standard class exercise, that number is being given to you by the user). Then transfer the scores from the string array returned by the Split method to the integer array you just created.

Submission Instructions

Submission should be made using a zip file that contains all of the Visual Studio C# project files. You will need to **zip the entire project folder** along with the .sln and .suo files. The folder will automatically contain the class source files as well as the executable file that is generated in \ProjectName\bin\Debug folder. Upload file to the Blackboard assignment drop box.

Zip the entire top-level folder by right-clicking the folder and selecting Send to | Compressed (zipped) folder.



Using built-in windows zip tools: <http://windows.microsoft.com/en-us/windows/compress-uncompress-files-zip-files>

Make sure you check the following. Your grade is dependent on all these criteria being met.

- You have included your name as a comment within your class.
 - e.g. “ // Inclass 5, Jane C. Smith, CIS 345, Tuesday 9:00 AM”
- Class file is called JaggedArrays.cs (rename from Program.cs).
- Your Visual Studio project is called JaggedArrayApp.
- Zip filename is: FirstNameLastName_Inclass5.zip

Verify your zip file before you submit

- Check for actual class files being present in the folder before you zip it.
- Check your zip file size after zipping – if it is 1K, it likely contains only a shortcut.
- Uncompress your zip file before submitting and verify that files are present.
- Download your zip file after submitting, uncompress, and again verify that your files are present.