



CIS 494 – Business Information Systems Development II – Spring 2015

Assignment 1

Due on Blackboard: Friday, January 30, by 11:59 PM

This assignment is meant to reacquaint you with basic structured programming concepts: creating methods, calling methods, creating arrays, passing arrays, etc. It also requires you to develop some basic logic.

Start a new project in Eclipse called *Assignment1* and create an *ArrayTest.java* file.

1) Java contains a method called *arraycopy* to enable copying of arrays. Implement a method of your own that provides the same functionality:

Create a method, *copyArray* that creates a copy of an array. This method should have two parameters, *sourceArray* and *targetArray*, both of which should be arrays of integers. It will not formally return anything (since arrays are references and do not need to be formally “returned.”)

Method Name: *copyArray*

Access modifier: Private

Parameters: 2 integer arrays, called *sourceArray* and *targetArray*.

Return value: none

Method Description

Loop through all the elements of the *sourceArray* and copy them into the *target array*.

The book contains most of the logic you would need to copy arrays in section 7.5 Copying Arrays. Section 7.6 shows how to pass arrays.

2) Create a method, joinArrays that concatenates or integrates values from two integer arrays into one array.

Method Name: joinArrays

Access modifier: Private

Parameters: 2 integer arrays, called array1 and array2

Return value: array of integers

Method Description

Within the method, you should create a new integer array called joinedArray, which is sized precisely to fit the elements of both the elements of array1 and array2. You should use your knowledge of each array's size.

Then you should go through both the arrays, one after the other, and copy all the elements into the joinedArray. All values should be copied. Be mindful that the index numbers will not match since you are trying to copy two sets of arrays into one array. So, part of your task is to work out a way to track index numbers for the joinedArray properly.

After you are doing copying the two arrays into the joinedArray, return it.

Book Section 7.6 shows how to pass arrays.

3) Create a method, generateRandoms that generates 100 random integers between 0 and 9 and displays the count for each number.

Method Name: generateRandoms

Access modifier: Private

Parameters: none

Return value: none

Method Description

You should use the Random class to create random numbers. Add an import statement for java.util.Random. Use the nextInt() method of the class, which allows you to specify an upperbound. *If you setup Eclipse correctly, "intellisense"-like behavior should inform you about method arguments and you should be able to see JavaDocs documentation as well.*

You do not need to store random numbers. Instead, you need to store how often you generated a random number from 0 to 9. Hint: use an array. Each element from 0 to 9 can store how many times each number was repeated. Your challenge is to come up with a way to increment the right element of the array.

Within a loop that repeats 100 times, generate a new random number between 0 and 10 (that is, up to 9) AND properly increment the right element in the array.

Finally, print a list of how many times each number was generated.

4) main method

Method Description

Within your main method, call each of the three methods you wrote and demonstrate that they work.

4.1) For calling the `copyArray()` method, first prepare two arrays, `testSourceArray` and `testTargetArray`. `testSourceArray` should be of size 5 and have as its values the numbers from 0 to 4.

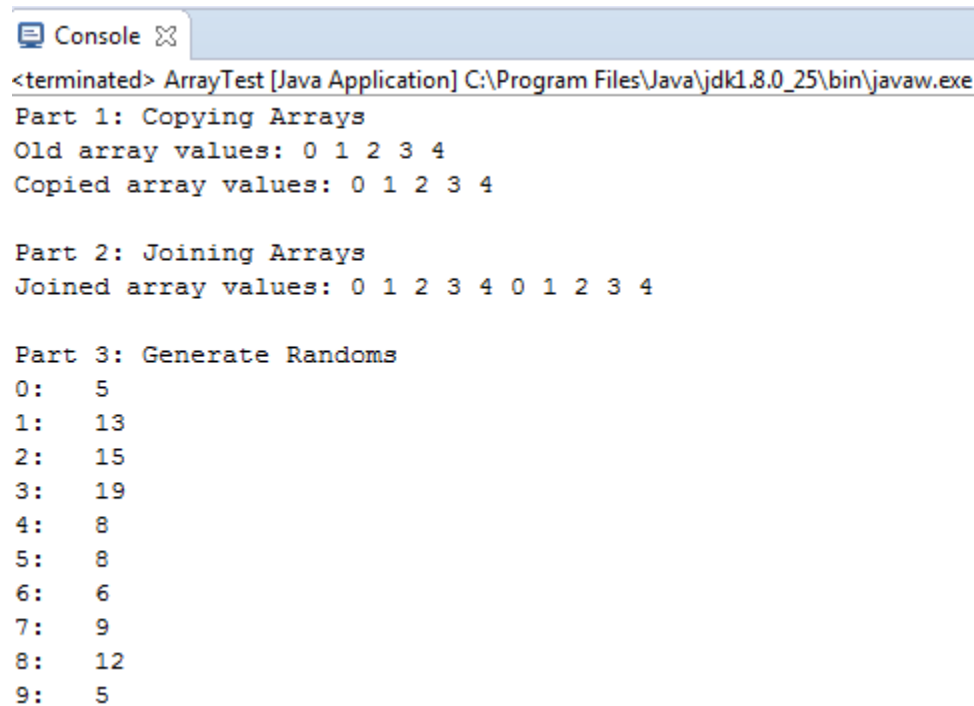
Call the `copyArray` method and provide it with the two arrays as arguments.

Then loop through both the `testSourceArray` and `testTargetArray` and print out its values.

4. 2) For calling the `joinArrays()` method, call the method, supply it with `testSourceArray` and `testTargetArray` as arguments. *If your `copyArray` works, `testTargetArray` would have the same values as `testSourceArray`, i.e 0 through 4.* Store the returned array in a new array of your choice and print out its values.

4.3) The call to `generateRandoms()` is simple and should not require any prep work.

Sample Output



```
<terminated> ArrayTest [Java Application] C:\Program Files\Java\jdk1.8.0_25\bin\javaw.exe
Part 1: Copying Arrays
Old array values: 0 1 2 3 4
Copied array values: 0 1 2 3 4

Part 2: Joining Arrays
Joined array values: 0 1 2 3 4 0 1 2 3 4

Part 3: Generate Randoms
0: 5
1: 13
2: 15
3: 19
4: 8
5: 8
6: 6
7: 9
8: 12
9: 5
```

Submitting Files

Submission should be made using a zip file that contains the entire Eclipse project folder. You will need to **zip the entire project folder**. The folder will automatically contain the class source files as well as the compiled .class files.

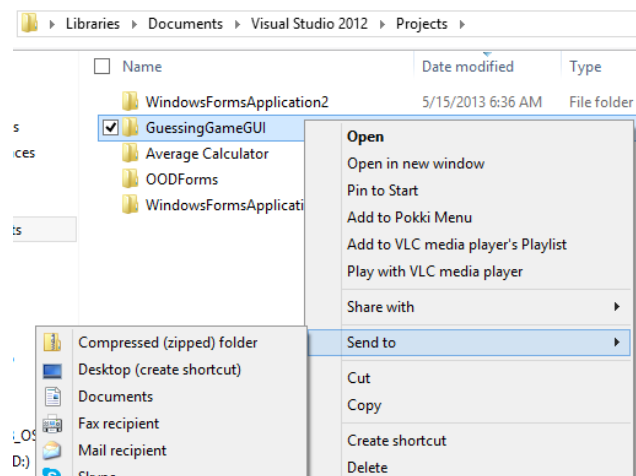
ZIP file should be named: **AX.zip or IC.zip**

where X is the in-class assignment number, e.g., A1.zip is the submission file for Assignment 1.

Note: The ZIP filename is independent of the Project name. Do not name your project A1.

How to Properly ZIP and Submit Your Project Files:

- Go to the folder within {Eclipse Workspace}\
- ZIP the entire top-level folder for your project by right-clicking your project folder and selecting Send to | Compressed (zipped) folder.
- Finally, submit the ZIP file using the submission link on Blackboard by the due-date and time listed on the assignment. Upload the ZIP file.



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

Verify your files BEFORE and AFTER submission:

- Check for actual class files being present in the folder before you zip it.
- ***Ensure that you are not zipping a short-cut to the folder.***
- After zipping, check file size. A file size under 4K likely does not contain all the files.
- Unzip, extract all files, and verify you see actual files, not a solitary short-cut.
- Uncompress your zip file before submitting and verify that files are present.
- Make sure you have submitted your file and not just saved a draft on Blackboard. ***A blue clock indicates a submission in progress, i.e. a draft, not a submission. The draft is accessible only to you. You will get a ZERO if you only ever save a draft on Blackboard and never submit your files.***
- Download your zip file after submitting, uncompress, and again verify that your files are present. Test your files in Eclipse after uncompressing.

This takes an extra couple of minutes. Please do it if your grade is important to you. If you do this, you will not end up submitting a bad file. If you submit an empty file, or one containing only a shortcut, or a bad zip file, or a bad project file, you will receive a score of zero and your only recourse will be to do the makeup assignment at the end of the semester.