
CSC172 LAB

JAVA COLLECTIONS

1 Introduction

The labs in CSC172 will follow a pair programming paradigm. Every student is encouraged (but not strictly required) to have a lab partner. Labs will typically have an even number of components. The two partners in a pair programming environment take turns at the keyboard. This paradigm facilitates code improvement through collaborative efforts, and exercises the programmers cognitive ability to understand and discuss concepts fundamental to computer programming. The use of pair programming is optional in CSC172. It is not a requirement. You can learn more about the pair programming paradigm, its history, methods, practical benefits, philosophical underpinnings, and scientific validation at http://en.wikipedia.org/wiki/Pair_programming .

Every student must hand in his own work, but every student must list the name of the lab partner (if any) in all lab reports.

This lab has six parts. You and your partner(s) should switch off typing each part, as explained by your lab TA. As one person types the lab, the other should be watching over the code and offering suggestions. Each part should be in addition to the previous parts, so do not erase any previous work when you switch.

The textbook should present examples of the code necessary to complete this lab. However, collaboration is allowed. You and your lab partner may discuss the lab with other pairs in the lab. It is acceptable to write code on the white board for the benefit of other lab pairs, but you are not allowed to electronically copy and/or transfer files between groups.

2 Some Java Utilities (import java.util.*)

The goal of this lab is to gain familiarity with the Java Collections Framework. Most of this, you will have to look up on line. You can copy the code directly off the on line documentation, but it will be considered cheating to give or receive an electronic copy of any source code files from another student.

1. Take a quick look at the documentation for the **Arrays** class (in `java.util.Arrays`). Write a small JAVA program that declares three integer arrays and one double. Declare a final integer value to use as the size of the arrays. Write static methods to print out the values of the arrays.
2. Write code to fill the double array with random doubles. Change one of the values in the array

to a known value. Print the array out. Use `Arrays.sort()` to sort the array. Print out the sorted array.

3. Write code to fill one of the integer arrays with random integers. Use `System.arraycopy()` to copy the integer array into another integer array. Print out both arrays.
4. Write code that uses `Arrays.fill()` to fill the third integer array with a single value. Print out the array.
5. Write code that uses `Arrays.equals()` to show that the first integer array is equal to the second. Write code that uses `Arrays.equals()` to show that the first integer array is not equal to the third.
6. Write code that uses `Arrays.binarysearch()` to look up the known value you inserted into the array of doubles (that is now sorted). Write code to demonstrate what happens when you try to use `Arrays.binarysearch()` to look up a value that is not contained in the array. Write code to demonstrate what happens when you use `Arrays.binarysearch()` to look up a value in an unsorted array (use the first or second array of integers).

3 Hand In

Hand in the source code from this lab at the appropriate location on the blackboard system at `my.rochester.edu`. You should hand in a single compressed/archived (i.e. “zipped”) file that contains the following.

1. A README (`172/grading.html`) that includes your contact information, your partner's name, a brief explanation of the lab (A one paragraph synopsis including information identifying what class and lab number your files represent.).
2. A plain text file named `OUTPUT` that includes author information at the beginning and shows the compile and run steps of your code. The best way to generate this file is to cut and paste from the command line.

4 Grading

`172/grading.html`

Each section (1-6) accounts for 15% of the lab grade (total 90%)

(README file counts for 10%)