

FIT2081

Laboratory for Week 04

Relates to Lecture 03

This Lab is one of 11 you will attend during the semester. Each lab is worth 4 of your final marks for the unit. The best 10 marks will be used.

In the last 30 minutes of the each lab (or before if you ask and the tutor has time) your tutor will mark you. You will be asked a few questions to determine whether you **confidently** understand the work you are presenting. You will be awarded a mark of 0 if **any** trace of plagiarism is detected during this questioning. You may prepare your solution before the lab in which case you can ask to be marked at the earliest opportunity.

Purpose:

- Increase the complexity of Java program we can write and include arrays, ArrayLists and method calling into our code.
- No practical examples of structured exception handling are included in this tutorial but this should not downplay its importance. We will run into examples later in Android programming.
- The difference between value and reference types will permeate all our code from now on. For instance Q3 relies on the side effects of passing arrays.

Important

- If you are not familiar with Scribble just treat the any presented Scribble scripts presented as pseudo code.
- Each question should be answered in its own class in its own project or within a common project. Please note the compiler compiles all classes in a project at once (this can be changed but not recommended for beginners)
- In any case use a single class with a main method as your basic Java plumbing for each question
- You will output to the IDE console so no pretty pencil pictures and word balloons or time limited displays (e.g. for 2 secs) but your Java output text should be the same as any presented Scribble output text. Line throws need not be duplicated.

Write Java code for each of the following Scribble scripts:

Q1 (1 mark) – Selection Sort

Write equivalent Java code to perform the same function as the presented Scribble code. Test your code to make sure it's sorting correctly.

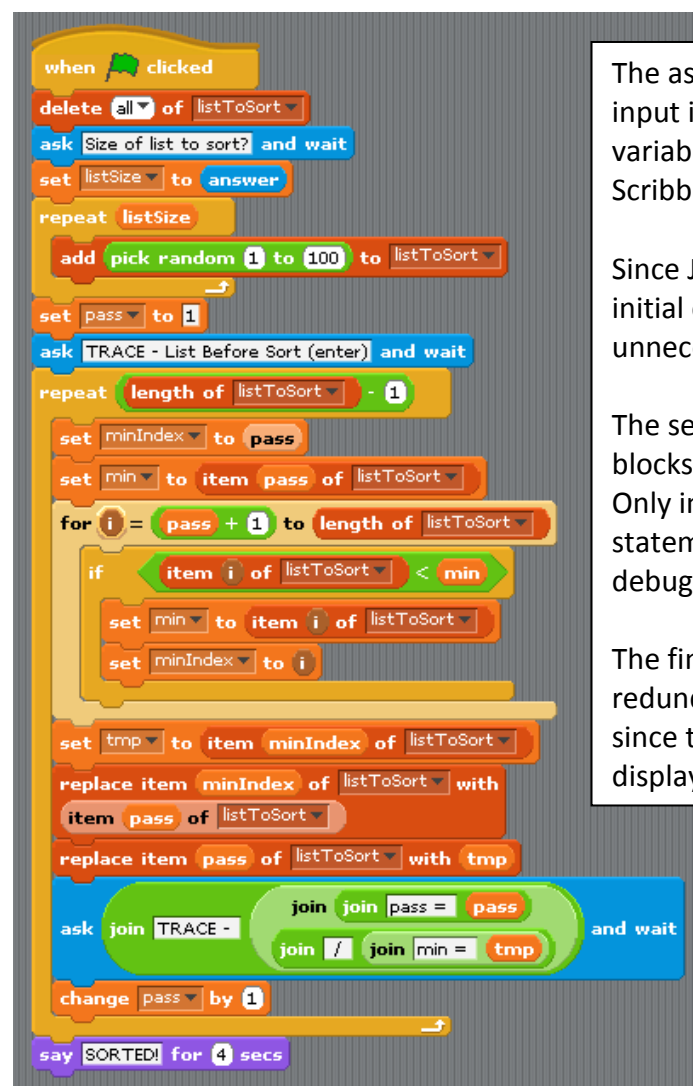
Notes:

- Use a Java array not an ArrayList
- Scribble array indexes are 1-based, Java array indexes are 0-based.
- Write a method in addition to main called displayArray(...). It should take a single parameter with type array of integer. It should display the elements of the passed array to the console on a single line. Elements should be comma separated. There should be no trailing comma after the last element. An example output appears below. This method should be called to display the array before and after sorting.

Size of array to sort?

20

75, 61, 24, 39, 82, 84, 24, 76, 24, 78, 62, 46, 75, 100, 87, 91, 92, 46, 70, 10
10, 24, 24, 24, 39, 46, 46, 61, 62, 70, 75, 75, 76, 78, 82, 84, 87, 91, 92, 100



The ask block assigns the input it collects to the answer variable automatically in Scribble.

Since Java is not “live” the initial clearing of the array is unnecessary

The second and third ask blocks are just for tracing. Only include equivalent Java statements if you need to debug.

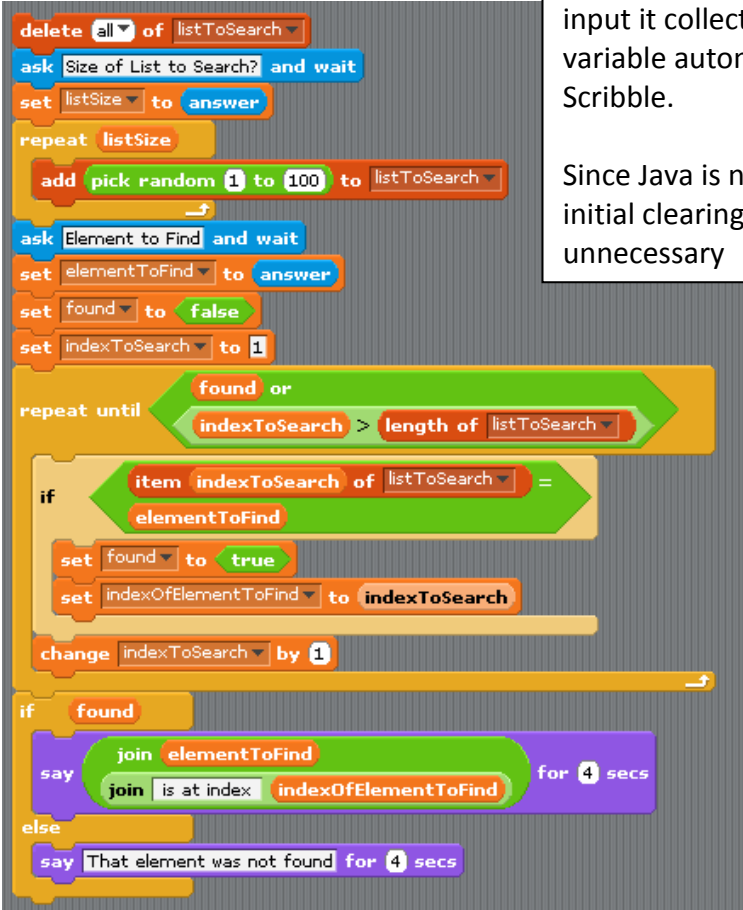
The final say block is redundant in the Java version since the sorted array is displayed (or not!)

Q2 (1 mark) – Linear Search

Write equivalent Java code to perform the same function as the presented Scribble code. Test your code to make sure it's searching correctly in both the found and not found case. Also test to find the first element and the last element and a normal element. You will need to trace the array before you search to do these tests. Use the debugger. Your tutor will expect you demonstrate your use of the debugger to perform these tests.

Notes:

- Scribble array indexes are 1-based, Java array indexes are 0-based.
- Use a Java array not an ArrayList.
- Be very careful with the search loops condition.
- If the compiler complains one of your variables may not have been initialised just initialise to a safe value in its declaration.



The ask block assigns the input it collects to the answer variable automatically in Scribble.

Since Java is not “live” the initial clearing of the array is unnecessary

Q3 (1 mark)

Code a Java **method** that accepts a String array and a String. The method should return true if the string can be found as an element of the array and false otherwise. Test your method by calling it from the main method which supplies its two parameters (no user input required). Use an array initialiser list to initialise the array you pass. Test thoroughly.

Notes:

- Your method should be private and static (explanations soon).
- Do not use the == operator to compare the contents of Strings (it compares their addresses).

Q4 (1 mark)

Code a Java **method** that accepts an ArrayList of integers and an integer. The method should delete all elements in the array list exactly divisible by the integer and return the number of remaining elements.

e.g. if the calling, main method supplies an ArrayList of integers with elements 1, 2, 3, ... , 100 and an integer of three your method should return the number of remaining elements i.e. the number of elements not divisible by three. It's 67 by the way.

Notes

- Call your method from main and test thoroughly with small arrays.
- Expect problems – think about what happens to the indexes of other elements when an element is removed
- For those that already know Java I do not want you to use an iterator