**Lab 1. Sorting I**

1. **Compiling Java programs (with command line)**
   1. Go to the directory containing the Java code documents (\*.java).
   2. Compile the Java code with the following command:

|  |
| --- |
| javac TargetClass.java |

* 1. Execute the class with main() by

|  |
| --- |
| java TargetClass |

1. **Sorting Algorithm and BigArray**
   1. You should have already encountered different sorting algorithms in previous programming course. In this lab, we will use Quicksort and Mergesort to compare the capacity of the sorting algorithm for PC and Raspberry Pi. Get yourself familiar with the principles of the sorting algorithms.

**Quicksort**: <https://en.wikipedia.org/wiki/Quicksort>

**Mergesort**: <https://en.wikipedia.org/wiki/Merge_sort>

* 1. Get yourself familiar with the given code:
     1. There is a class *BigArray* that represents large arrays containing big integers (long). You can specify the size of it by passing an integer to the constructor. For example, to initialize a *BigArray* that can contain 20 elements, you can do:

|  |
| --- |
| BigArray big\_array = new BigArray(20); |

* + 1. To fill the array with random integers, you can call

|  |
| --- |
| big\_array.initRandomCase(); |

* + 1. Since the elements are randomly generated, it is probably not ordered in the array. The class support two sorting algorithms to put the elements in order:

|  |
| --- |
| big\_array.quicksort(); // To perform Quicksort on the array  big\_array.mergesort(); // To perform Mergesort on the array |

* + 1. To verify the array is sorted, you can use the method:

|  |
| --- |
| big\_array.isSorted(); // return true if the array is sorted |

1. Capacity of BigArray
   1. Let us try to use BigArray to store and sort elements. Try to execute the following code in the main() of BigArray. What is the output of the program?

|  |
| --- |
| int size = 100;  BigArray big\_array = new BigArray(size);  big\_array.mergesort();  big\_array.isSorted(); |

* 1. Now, change the size of the BigArray to 1000000000. What is the output?
  2. Change the size of the BigArray to 100000000. What is the output again? What is the difference between the output in 3.2 and 3.3?
  3. The BigArray will be used to store and sort large arrays on different platform (i.e. on your lab computer and Raspberry Pi). To understand the capacity of the class, we would like to design a program to find out the maximum elements that it can store and sort.
     1. In main(), design a program that can find the approximate maximum number of elements can be stored in BigArray. Execute the program on PC and Raspberry Pi and compare the result.
     2. Modify the program to find the approximate maximum number of elements that can be sorted using BigArray with mergesort(). Execute the program on PC and Raspberry Pi. Comparing to the number obtained in 3.4.1, suggest reasons that lead to the difference of capacity (if any).

Raspi: max cap 10,116,005

PC: max cap 87,982,393

\*\* parallel sort -> large number

* Serializable error?

Bluetooth