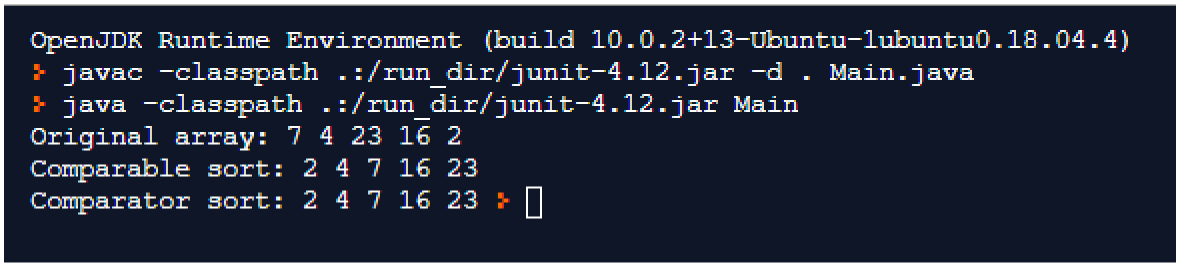
Sean Poston

CS300 Homework 4 – 23.8

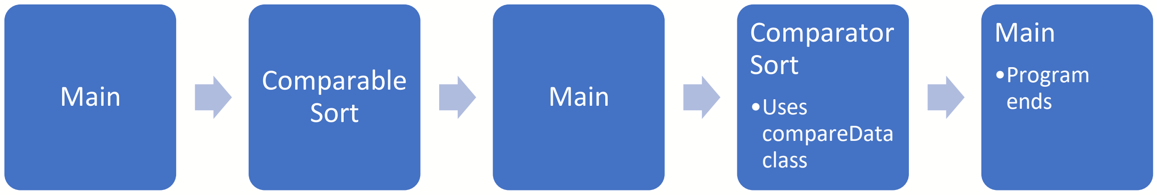
2/27/2020

https://repl.it/@seanposton4/CS300-238

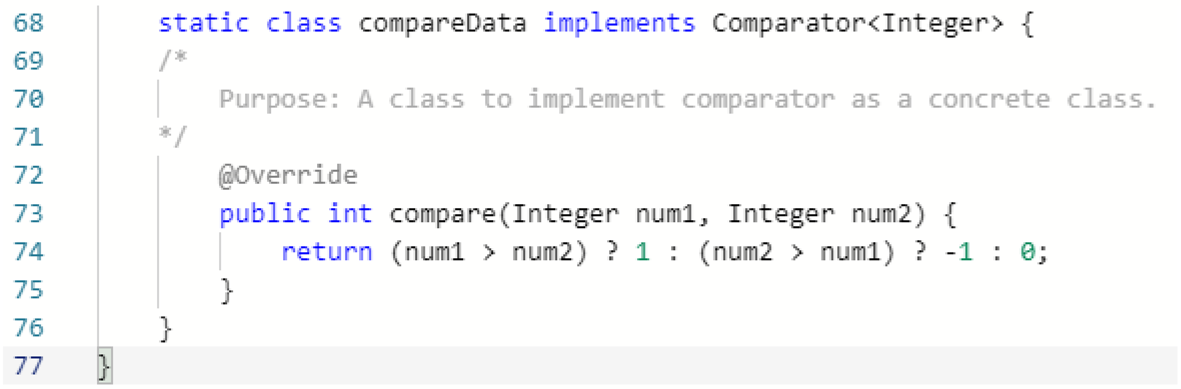


```
OpenJDK Runtime Environment (build 10.0.2+13-Ubuntu-1ubuntu0.18.04.4)
› javac -classpath .:/run_dir/junit-4.12.jar -d . Main.java
› java -classpath .:/run_dir/junit-4.12.jar Main
Original array: 7 4 23 16 2
Comparable sort: 2 4 7 16 23
Comparator sort: 2 4 7 16 23 ›
```

This is the output of the program. The program starts with an unsorted array of randomly chosen elements. These are then sent to the first sort method which uses the built in comparable "compareTo" method. The program then returns to main after printing this sorted array and is sent to the second method which uses the Comparator method.



Since Comparator is an interface, it's unable to be constructed on its own, which is why the static class compareData is used. This class has a single method which defines the rules for which to sort the data. Simply put, if number 1 is greater than number 2, return 1. Return 2 if vice-versa and return 0 if they're the same number. These rules are used in sorting the array.



```
    static class compareData implements Comparator<Integer> {
    /*
        Purpose: A class to implement comparator as a concrete class.
    */
        @Override
        public int compare(Integer num1, Integer num2) {
            return (num1 > num2) ? 1 : (num2 > num1) ? -1 : 0;
        }
    }
}
```

After this is run, the program will return to the main function and end because there is nothing else to do.

```
/*
Author: Sean Poston
Purpose: To implement two different sorting arrays using comparator and
comparable.
Date: 2/27/2020
*/

import java.util.*;
class Main {
  public static void main(String[] args) {
    Integer[] list = {7, 4, 23, 16, 2}; //Integer array initialization
    System.out.print("Original array: ");
    for (int i = 0; i < list.length; i++)
        System.out.print(list[i] + " "); //Print unsorted array.

    System.out.print("\nComparable sort: ");
    insertionSort(list); //Send to comparable sort method.

    System.out.print("\nComparator sort: ");
    insertionSort(list, new compareData()); //Send to comparator sort method.
  }

  public static <E extends Comparable<E>> void insertionSort(E[] list) {
      /*
        Purpose: To sort an array of type <E extends Comparable<E>> that will be
insertion sorted with the help of comparable.
        Precondition: Must have an array of a type that extends Comparable.
        Postcondition: Will print out the sorted version of the array.
      */
    int j;
    E temp;

    for (int i = 1; i < list.length; i++) {
      j = i;
      while (j > 0 && list[j - 1].compareTo(list[j]) > 0) {
        temp = list[j - 1];
        list[j - 1] = list[j];
        list[j] = temp;
        j--; //Nested loops to sort the array.
      }
    }

    for (int i = 0; i < list.length; i++)
      System.out.print(list[i] + " ");
  }
  public static <E> void insertionSort(E[] list, Comparator<? super E>
comparator) {
      /*
        Purpose: To sort an array of type <E> that is sorted using the
specifications defined in compareData (Line 69).
        Precondition: Must have an array of type E and a comparator from the
compareData class.
        Postcondition: Will sort the given array and print out the array.
      */
    int j;
    E temp;

    for (int i = 1; i < list.length; i++) {
      j = i;
      while (j > 0 && comparator.compare(list[j - 1], list[j]) > 0) {
        temp = list[j - 1];
        list[j - 1] = list[j];
        list[j] = temp;
        j--; //Nested loop to sort the array.
      }
    }

    for (int i = 0; i < list.length; i++)
      System.out.print(list[i] + " ");
  }

    static class compareData implements Comparator<Integer> {
    /*
        Purpose: A class to implement comparator as a concrete class.
    */
        @Override
        public int compare(Integer num1, Integer num2) {
            return (num1 > num2) ? 1 : (num2 > num1) ? -1 : 0;
        }
    }
}
```