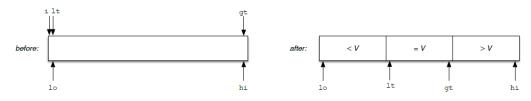
COS 226 Data Structures and Algorithms Computer Science Department Princeton University Spring 2016

Week 3 Activity

1. 3-way Partitioning

Apply Dijkstra's 3-way partitioning algorithm to partition (not sort) the array below. This algorithm groups elements into three groups as shown here, where v is the current pivot element.



Recall that in this algorithm:

- Let v be partitioning item which is initially a[lo] (v never changes afterwards).
- Scan i from left to right.
 - (a) (a[i] < v): exchange a[lt] with a[i]; increment both lt and i
 - (b) (a[i] > v): exchange a[gt] with a[i]; decrement gt
 - (c) (a[i] == v): increment *i*.

The algorithm terminates when i > gt. The 3-way partitioning algorithm is applied to the array below: write its contents after each exchange. (Consider that every occurrence of R_i is the same letter; the indices are just provided to clarify which R's are involved in each exchange.)

R ₁	R_2	E	Α	L	S	0	R_3	R ₄	Т

2. Comparators Algorithms Textbook 2.5

return ____;

- (a) If a Java class implements the Comparable interface, what method must be provided in the class code?
- (b) If a Java class implements the Comparator interface, what method must be provided in the class code?
- (c) Consider the algs4 Point2D class provided in the appendix (see precept page for complete code). Suppose we want to create a Comparator that compares two Point2D objects based on their distance from some third point. Fill in the code below. The compare must return 1, 0 or -1 if point 1 is more, same or less distance from point 2. You may find the code in the appendix helpful.

```
public static class DistanceComparator implements Comparator<____> {
   Point2D _____;
   public DistanceComparator(_____) {
     _____ = ____;
   }
   public int compare(_____ p, _____ q) {
     double distToP = p.distanceTo(_____);
     double distToQ = q.distanceTo(_____);
     if (distToP < distToQ) return ____;
     if (distToP > distToQ) return ____;
   }
}
```

```
}
```

}

(d) Now suppose we want to use our comparator to sort an array of Point2D objects by their distance from the origin. Fill in the code below to accomplish this task.

```
Point2D[] points = getRandomPoints();
Point2D origin = new Point2D(_____);
Comparator<Point2D> originDistanceComparator = _____;
Arrays.sort(points, ______);
```

(e) When Arrays.sort() is called, which sorting algorithm does Java use to sort the array?

3. Sorting Invariants

The column on the left is the original input of strings to be sorted; the column on the right are the strings in sorted order; the other columns are the contents at some intermediate step during one of the 6 sorting algorithms listed below. Match up each algorithm by writing its number under the corresponding column. Use each number exactly once.

nite rein deni dent rent ding grin ride rind diet dint ring dire dreg edit	deni dent nite rein ding grin rent ride diet dint rind ring dine dire dreg	deni dent ding grin nite rein rent ride diet dint rind ring dine dire dreg	deni dent ding nite rein rent grin ride rind diet dint ring dire dreg edit	deni diet dine ding rent dint ride rind nite grin ring dire dreg edit	dint deni dent edit ding grin dreg dire diet nite ring rind ride rent	dine deni dent edit ding grin dreg dire diet dint nite ring rind ride rent	deni dent diet dine ding dint dire dreg edit grin nite rein rent ride rind
edit dine	dreg edit	dreg edit	edit dine	edit rein	rent rein	rent rein	rind ring
00	Care	- are				rem	i iiig

(0) Original input

(4) Mergesort (top-down)

- (6) Quicksort (standard, no shuffle)
- (5) $\frac{\text{Mergesort}}{(bottom-up)}$

(7) Quicksort (3-way, no shuffle)

(2) Selection sort(3) Insertion sort

(1) Sorted

4. 3-way Merge sort (Bonus Question)

3-way merge sort is a modification of the merge sort algorithm that considers 3 "equal" sub arrays instead of 2 sub arrays.

- (a) Given 3 sorted sub arrays of size N/3, how many comparisons are needed to merge them to a sorted array of size N. Provide your answer in tilde notation.
- (b) Argue that number of compares to sort an array of size N using 3-way merge sort is still linearithmic.
- (c) Given a choice, would you choose 3-way or 2-way merge sort? Justify your answer.

1 Appendix

```
Below is the syntax highlighted version of Point2D.java from Algorithms.
```

```
* Compilation: javac Point2D.java

    Immutable point data type for points in the plane.

 import java.util.Arrays;
import java.util.Comparator;
public class Point2D implements Comparable<Point2D> {
    public static final Comparator<Point2D> X_ORDER = new XOrder();
 // see precept page for full code
    \ensuremath{\prime\prime}\xspace by y-coordinate, breaking ties by x-coordinate
    public int compareTo(Point2D that) {
    if (this.y < that.y) return -1;
    if (this.y > that.y) return +1;
    if (this.x < that.x) return -1;</pre>
         if (this.x > that.x) return +1;
         return 0;
    3
    // compare points according to their x-coordinate
private static class XOrder implements Comparator<Point2D> {
         public int compare(Point2D p, Point2D q) {
             if (p.x < q.x) return -1;
if (p.x > q.x) return +1;
             return 0;
        }
    }
   // see precept page for full code
    // does this point equal y?
    public boolean equals(Object other) {
    if (other == this) return true;
         if (other == null) return false;
         if (other.getClass() != this.getClass()) return false;
Point2D that = (Point2D) other;
         return this.x == that.x && this.y == that.y;
    Ъ
    // see precept page for full code
    public static void main(String[] args) {
        // code not given
    }
}
```

Copyright 2002..2010, Robert Sedgewick and Kevin Wayne. Last updated: Tue Oct 18 07:05:59 EDT 2011.