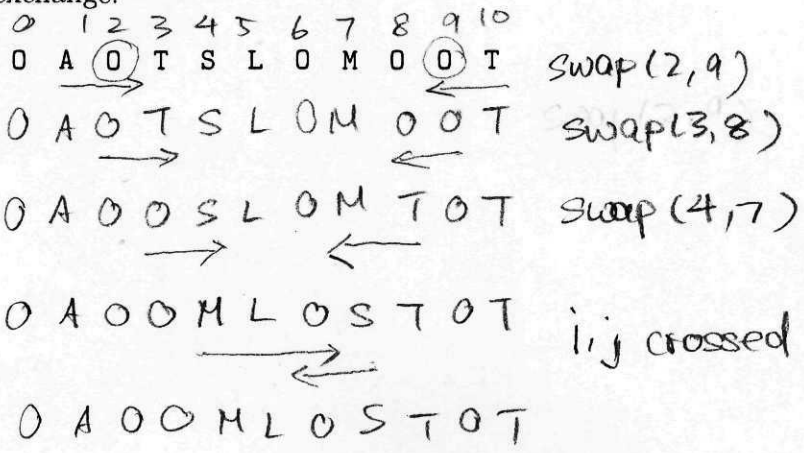
COS226 Week 3 Activity

1. *Quicksort. Algorithms textbook 2.3*

   Suppose that the result of the shuffle in Algorithm 2.5 is `O A O T S L O M O O T`. Show the result of the first call to `partition()` by giving the contents of the array after each exchange.



2. *Static Comparators. Algorithms textbook 2.5*

   (a) Given an array of N `Point2D` objects, call it `pts`, describe a linearithmic algorithm to remove all duplicates. *Hint:* sort.

   1. Step sorting O(Nlog N)
      Important to use stable sort so that each following sort doesn't mess up the relative order of the previous sort.
   2. Scan thru array to find duplicates O(N)

   (b) Suppose you are sorting Point2D objects by using: `Arrays.sort(pts);` Which comparison method in Point2D is used?

   compareTo()

   (c) Write a Java code fragment that sorts using one or more of the static comparators defined in `Point2D (X_ORDER, Y_ORDER, R_ORDER)`

   Arrays.sort(pts, Point2D.X_ORDER)

   (d) For each sort you used in the previous question show the order of the following points after that sort takes place:
   A(4, 5) B(3, 5) C(4, 2) D(3, 2) X order : B D A C

3. *Dynamic Comparators. Algorithms textbook 2.5* Consider the following code.

```
public class Point2D
{
   public final Comparator<Point2D> POLAR_ORDER = new PolarOrder();
   private final double x, y;
   ...
   private static int ccw(Point2D a, Point2D b, Point2D c)
   { /* see lecture slides or booksite */ }

   private class PolarOrder implements Comparator<Point2D>
   {
      public int compare(Point2D q1, Point2D q2)
      {
         double dx1 = q1.x - x;
         double dy1 = q1.y - y;
         double dx2 = q2.x - x;
         double dy2 = q2.y - y;
         if (dy1 == 0 && dy2 == 0) { ... }          // see question (c)
         else if (dy1 >= 0 && dy2 < 0) return -1;
         else if (dy2 >= 0 && dy1 < 0) return +1;
         else return -ccw(Point2D.this, q1, q2);
      }
   }
}
```

(a) What is the difference between a `Comparable` and a `Comparator`?

Comparable: compareTo(),
Comparator: compare() with different Comparators, can sort one object array via different criteria.

Example: Arrays.sort(a) to sort String uses compareTo(), but Arrays.sort(a, String.CASE_INSENSITIVE_ORDER)

(b) What is the difference between a static and a dynamic `Comparator`?

Point2D.java: first 3 are static, next 3 dynamic Comparators. usesComparator
static only compares two arguments passed thru compare()
dynamic can additionally compare with the object that's calling it.

(c) Consider the special case code in the compare method. What does ccw() calculate when dy1 or dy2 is 0? If the line with the question (c) comment did not exist what would happen when `-ccw()` is called?

ccw: counterclockwise, see Point2D
returns 0 when dy1 dy2 0.
ccw is calculating a different function, therefore we need to add some special cases that it doesn't handle.