Name : zhenzhen Su

Class: CSC 143

Instructor: Dr. Reid

Problem 1: what is index for middle element and what value will be returned?

a. 103 index : 4,7,8,9
value: 9

b. 30 index : 4,7,5,6
value: -7

c. 8 index: 4,1
value: 1

d. -1 index: 4,1,0
value -2

Problem 2: what is index for middle element and what value will be returned?

a. 13 index: 5, 2, 3, 4
value -5

b. 39 index: 5,8,6,7
value: 7

c. 50 index:5,8,10
value: 10

d. 2 index: 5,2,0,1
value: -2

Problem 4: To which complexity class is the algorithm belong to? Explain it.

```
Public static int[] mystery1( int[] list){
        Int[] result = new int[2* list.length];  (twice the length of input array)
        For (int I = 0; I < list.length; i++){
                ...
        }
```

Answer: O(N) because the for loop will be execute for N times even though the total times is 2*N + 1. We pick the term with highest power.

Problem 12: state of elements after each pass of outmost loop

```
Int[] numbers5 = {22,44,11,88,66,33,55,77}
{11,44,22,88,66,33,55,77}
{11,22,44,88,66,33,55,77}
{11,22,33,88,66,44,55,77}
{11,22,33,44,66,88,55,77}
{11,22,33,44,55,88,66,77}
{11,22,33,44,55,66,88,77}
{11,22,33,44,55,66,77,88}

Int[] numbers6 = {-3,-6,-1,-5,0,-2,-4,-7}
```

```
{-7,-6,-1,-5,0,-2,-4,-3}
{-7,-6,-1,-5,0,-2,-4,-3}
{-7,-6,-5,-1,0,-2,-4,-3}
{-7,-6,-5,-4,0,-2,-1,-3}
{-7,-6,-5,-4,-3,-2,-1,0}
{-7,-6,-5,-4,-3,-2,-1,0}
{-7,-6,-5,-4,-3,-2,-1,0}
```

Problem 13: Problem 12: state of elements after each pass of outmost loop

```
Int[] numbers5 = {22,44,11,88,66,33,55,77}
Sort {22,44,11,88}

Sort{22,44}

Sort{22}

Sort{44}

Merge{22} and {44}

Sort{11,88}

Sort{11}

Sort{88}

Merge{11} and {88}

Merge{22,44} and {11,88}

Sort{66, 33,55,77}

Sort{66,33}

Sort{66}

Sort{33}

Merge{66} and {33}

Sort{55,77}

Sort{55}

Sort{77}

Merge{55} and {77}

Merge{33,66} and {55,77}

Merge{11,22,44,88} and {33,55,66,77}

{11,22,33,44,55,66,77,88}



Int[] numbers6 = {-3,-6,-1,-5,0,-2,-4,-7}
Sort{3,-6,-1,-5}
Sort{3,-6}
Sort{3}
Sort{-6}
Merge{3} and {-6}
Sort{-1,-5}
Sort{-1}
```

```
Sort{-5}
Merge{-1} and {-5}
Merge{-6,3}and {-5,-1}
Sort{0,-2,-4,-7}
Sort{0,-2}
Sort{0}
Sort{-2}
Merge{0} and {-2}
Sort{-4,-7}
Sort{-4}
Sort{-7}
Merge{-4} and {-7}
Merge{-2,0} and {-7,-4}
Merge{-6,-5,-1,3} and {-7,-4,-2,0}

{-7,-6,-5,-4,-1,0,3}
```

Problem 15: Write a comparator that compares point objects by their distance from origin. Points that are closer to origin are considered to come before points that are farther from origin.

Code:

```
public class PointComparator implements Comparator<Point> {
    public int compare(Point p1, Point p2){
        double dist1 = Math.hypot(p1.getX(), p1.getY());
        double dist2 = Math.hypot(p2.getX(), p2.getY());
        if(dist1 == dist2){
            return 0;
        }else if(dist1 < dist2){
            return -1;
        }else{
            return 1;
        }
    }

}


public class Main {

    public static void main(String[] args) {
   // write your code here 4/12/13/15
        Point point1 = new Point(2,4);
        Point point2 = new Point(-2,3);
        Point[] pointArray = {point1, point2, new Point(-1,2)};
        Arrays.sort(pointArray, new PointComparator());
        System.out.println(Arrays.toString(pointArray));
    }
}
```

Ouput Console:

[java.awt.Point[x=-1,y=2], java.awt.Point[x=-2,y=3], java.awt.Point[x=2,y=4]]