알고리즘 14주차 과제 보고서

충남대학교 컴퓨터공학과
알고리즘 04분반
학번: 201701975
이름: 구건모

```java
package model;

public class Point {
    public int x;
    public int y;

    public Point(int givenX, int givenY) {
        this.x = givenX;
        this.y = givenY;
    }

    public Point() {
        this(0, 0);

    }

    public long distanceTo(Point other) {
        if(other == null) {
            return Integer.MAX_VALUE;
        }
        else
        {
            long differenceOfX = this.x - other.x;
            long differenceOfY = this.y - other.y;
            return (differenceOfX*differenceOfX + differenceOfY*differenceOfY);

        }
    }


}

```

Class Point

Point 클래스는 한 점에 대한 정보를 가진 클래스로, x,y 좌표를 Instance 변수로 가지고 자신으로부터 다른점까지의 거리를 연산하는 distanceTo 메서드가 구현되어 있습니다.

```java
package model;

import list.ArrayList;

public class PointSet extends ArrayList<Point> {
    public PointSet(int givenCapacity) {
        super(givenCapacity);
    }

    public Point pointReferenceByIndex(ReferenceList referenceList, int Index) {
        if (referenceList.orderIsValid(Index)) {
            return this.elementAt(referenceList.elementAt(Index));

        } else {
            return null;
        }
    }
}

```

Class PointSet

PointSet은 포인트들을
담기 위한 자료구조이며
ReferenceList 로 저장되어있는
포인트들의 실제 값들을 찾기 위한 메서드
도 구현되어 있습니다.

```java
package model;

import list.ArrayList;

public class ReferenceList extends ArrayList<Integer> {
    public ReferenceList(int givenCapacity) {
        super(givenCapacity);
    }
}
```

Class ReferenceList

이번 구현에서는 입력받은 PointSet을 Q, R 로 Divide 할 때에 PointSet의 Index 값을 참조하여 저장하는데 이때 사용하게될 ReferenceList 입니다.

ArrayList로 구현됩니다.

```java
package model;

public abstract class ReferenceListOrderedByCoordinate extends ReferenceList {

    private PointSet _pointSet;

    protected PointSet pointSet() {
        return this._pointSet;
    }

    private void setPointSet(PointSet newPointSet) {
        this._pointSet = newPointSet;
    }

    public ReferenceListOrderedByCoordinate(PointSet givenPointSet) {
        super(givenPointSet.size());
        this.setPointSet(givenPointSet);
        this.generateReferenceListOrderedByCoordinate();

    }

    protected abstract int coordinateReferencedByIndex(int i);

    private void swap(int i, int j) {
        Integer temp = this.elementAt(i);
        this.setElementAt(this.elementAt(j), i);
        this.setElementAt(temp, j);
    }

    private int compareCoordinate(int i, int j) {
        int coordinateReferenceBy_i = this.coordinateReferencedByIndex(i);
        int coordinateReferenceBy_j = this.coordinateReferencedByIndex(j);

        if (coordinateReferenceBy_i < coordinateReferenceBy_j) {
            return -1;
        } else if (coordinateReferenceBy_i > coordinateReferenceBy_j) {
            return +1;
        } else {
            if (this.elementAt(i) < this.elementAt(j)) {
                return -1;
            } else if (this.elementAt(i) > this.elementAt(j)) {
                return +1;
            } else {
                return 0;
            }
        }
    }

    private int pivotByMedian(int left, int right) {
        int mid = (left + right) / 2;
        if (this.compareCoordinate(left, mid) < 0) {
            if (this.compareCoordinate(mid, right) < 0) {
                return mid;
```

class ReferenceListOrderedByCoordinate

해당 클래스는 임의의 pointSet을 받아서 QuickSort를 이용하여 정렬을 수행하도록 하는 기능을 가지고 있는 추상 클래스로, 이후에 ReferenceListOrderedByCoordinateX, ReferenceListOrderedByCoordinateY 클래스에서 각각 Extends 하여 사용합니다.

```java
package model;

public class FindClosestPair {
    private class SeparatedPair {
        private ReferenceList _leftList;
        private ReferenceList _rightList;

        public SeparatedPair() {

        }
    }

    private PairOfPoints closestPairDirectlyFromSmallPointSet(ReferenceList Px) {
        if (Px.size() <= 1) {
            return null;
        } else {
            Point p0 = this.pointSet().pointReferenceByIndex(Px, 0);
            Point p1 = this.pointSet().pointReferenceByIndex(Px, 1);

            Point closestPair_point_i = p0;
            Point closestPair_point_j = p1;
            long minDistance = p0.distanceTo(p1);

            if (Px.size() == 3) {
                Point p2 = this.pointSet().pointReferenceByIndex(Px, 2);
                if (p0.distanceTo(p2) < minDistance) {
                    closestPair_point_i = p0;
                    closestPair_point_j = p2;
                    minDistance = p0.distanceTo(p2);

                }
                if (p1.distanceTo(p2) < minDistance) {
                    closestPair_point_i = p1;
                    closestPair_point_j = p2;
                }
            }
            PairOfPoints closestPair = new PairOfPoints(closestPair_point_i, closes
            return closestPair;
        }
    }

    private Integer seperationLine(ReferenceList Px) {
        return Px.elementAt(Px.size() / 2);
    }

    private PairOfPoints solveRecursively(ReferenceList Px, ReferenceList Py) {
        PairOfPoints closestPair;
        if (Px.size() <= 3) {
            closestPair = this.closestPairDirectlyFromSmallPointSet(Px);
            return closestPair;

        }
        int seperationLine = this.seperationLine(Px);
```

Class FindClosestPair

FindClosestPair는 ClosestPair를 찾는 알고리즘이 구현되어 있는 클래스로, Divide-and-Counquer 방식과 ComparingAllPairs 두가지 방식으로 찾을 수 있도록 구현되어 있습니다.

결과화면

```
_Main_AL14_201701975_구건모 [Java Application] C:₩Users₩gmku1₩.p2₩pool₩plugins₩org.e
<<< 최단거리 쌍 찾기를 시작합니다 >>>

?? 문제를 풀려면 'Y' 또는 'y', 아니면 다른 아무거나 치시오: y

[문제 풀이를 시작합니다]
? 점의 개수를 입력하시오: 5
! 점의 (X,Y) 좌표를 차례로 입력해야 합니다. ? X 좌표 값을 입력하시오: 1
? Y 좌표 값을 입력하시오: 1
! 점의 (X,Y) 좌표를 차례로 입력해야 합니다. ? X 좌표 값을 입력하시오: 5
? Y 좌표 값을 입력하시오: 2
! 점의 (X,Y) 좌표를 차례로 입력해야 합니다. ? X 좌표 값을 입력하시오: 4
? Y 좌표 값을 입력하시오: 10
! 점의 (X,Y) 좌표를 차례로 입력해야 합니다. ? X 좌표 값을 입력하시오: 8
? Y 좌표 값을 입력하시오: 3
! 점의 (X,Y) 좌표를 차례로 입력해야 합니다. ? X 좌표 값을 입력하시오: 0
? Y 좌표 값을 입력하시오: 9
! 점들의 집합입니다:
(1,1)
(5,2)
(4,10)
(8,3)
(0,9)
! Divide-and-Conquer 방식으로 찾은 최단거리 쌍:<(5,2), (8,3)>  Distance = 10
! 모든 쌍의 거리를 비교하여 찾은 최단거리 쌍:<(5,2), (8,3)>  Distance = 10

?? 문제를 풀려면 'Y' 또는 'y', 아니면 다른 아무거나 치시오:
```

Inheritance / interface: 어떻게 하면 안정적이고 효율적인 설계가 될 수 있는지?

이번 과제와 같이 interface를 통해서 구현해야 할 부분을 명시하여 구현할 수 있고, Inheritance를 이용하여 구현하게 되면 공통적인 부분들을 구현하지 않아도 되기 때문에 효율적인 설계가 된다고 생각합니다.

ReferenceListOrderedByCoordinate를 Abstract Class로 선언한 이유는?

ReferenceListOrderdByCoordinate는 각각 X,Y에 대해서 정렬을 할 때에 사실상 동일 한 기능을 수행하기 때문에 더 추상화된 ReferenceListOrderdByCoordinated 를 이용하여 ReferenceListOrderdByCoordinateX , ReferenceListOrderdByCoordinateY 로 구현하도록 선언하였습니다.

Instance variable을 private으로 선언한 일반적인 이유는?

일반적으로 한 클래스내의 Instance variable은 그 클래스 내부에서만 사용하는 경우가 대부분이기 때문에 외부에서 임의로 Instance 변수들에 대한 수정이 일어나지 않도록 하기 위해서 Private으로 접근을 제한합니다.

그럼에도 불구하고, class Point의 instance variable을 public으로 선언한 이유는?

point 들의 x와 y좌표를 직접 사용하는 빈도가 많다 보니 public으로 선언한 것이라고 생각합니다.