1.

interface Circle {

Point getPoint();

double getRadius();

boolean containedInCircle(Point point);

}

public class CircleImpl extends Circle {

private double x, y, radius;

public CircleImpl ( double x, double y, double radius){

this.x = x ;

this.y = y;

this.raduis = radius;

public getPoint(){

returns new point (this.x, this,y);

}

public getRadius(){

returns this.radius;

}

Boolean containedInCircle(Point point){

If( Point.getX <= x + radius && Point.getX >= x - radius){

If ( Point.getY <= y + radius && Point.getY >= x – radius){

Return true;

}

else return false;

else return false;

}

}

Main{

ArrayList circles = new Arraylist of circles;

Circles.add(new CircleImp(1,1,1);

Circles.add(new CircleImp(2,3,4);

(…)

Point search = new Point(some x, some y);

For( every circle in Arraylist)

{

if( circles.containedInCircle(search){

print that circle out;

}

}

}

The growth rate for this method would be O(n).