

# 最小圆覆盖

## Minimum Enclosing Circle

**Idea:** 随机增量法。假设圆  $O$  是前  $i - 1$  个点的最小覆盖圆，若第  $i$  个点在圆内或圆上，不用处理；否则，新圆应该经过第  $i$  个点。我们现在以第 1 个点和第  $i$  个点为直径作圆，但是这个圆可能包含不了所有前  $i - 1$  个点，设第  $j$  个点在圆外，那么新圆一定经过第  $j$  个点，以第  $i$  个点和第  $j$  个点为直径作圆，同理可得到新圆一定经过第  $k$  个点。三点确定一个圆，我们就找到了覆盖前  $i$  个点的最小圆。随机化以避免最坏情形出现。

**Complexity:** 期望  $O(n)$

**Code:**

```
1 Point get0(Point a, Point b, Point c){
2     Point mab = (a + b) / 2, mbc = (b + c) / 2;
3     Vector vab = Normal(b - a), vbc = Normal(c - b);
4     return GetLineIntersection(Line(mab, vab), Line(mbc, vbc));
5 }
6 Circle minEnclosingCircle(int n, Point p[]){
7     random_shuffle(p+1, p+n+1);
8     Point o = p[1]; double r = 0;
9     for(int i = 1; i <= n; i++){
10         if(cmp(Length(o - p[i]), r) <= 0) continue;
11         o = (p[i] + p[1]) / 2;
12         r = Length(p[i] - p[1]) / 2;
13         for(int j = 1; j < i; j++){
14             if(cmp(Length(o - p[j]), r) <= 0) continue;
15             o = (p[j] + p[i]) / 2;
16             r = Length(p[j] - p[i]) / 2;
17             for(int k = 1; k < j; k++){
18                 if(cmp(Length(o - p[k]), r) <= 0) continue;
19                 o = get0(p[i], p[j], p[k]);
20                 r = Length(p[i] - o);
21             }
22         }
23     }
24     return Circle(o, r);
25 }
```