《算法分析与设计》第 2 次作业 *

姓名：谈金翰　　学号：71118314

2020/3/15

算法分析题

题目 **1** ：求下列递推关系表示的算法复杂度

(1) $T(n) = 9T(n/3) + n$

(2) $T(n) = n + 3T(\frac{n}{4})$

(3) $T(n) = 4T(n/2) + n^2 \lg n$

答：

(1)$T(n)=9T(n/3)+n=n + 3n + 9n + ... + 3^k n \; =n(\frac{1-3^(\log_3 n)}{(1-3)})=O(n^2)$

(2)$T(n)=n+3T(\frac{n}{4})=n(1 + \frac{3}{4} + \frac{3^2}{4^2} + ... + \frac{3^k}{4^k})=n\frac{(1-(\frac{3}{4})^k)}{1-\frac{3}{4}} \; =4n=O(n)$

(3)$T(n)=n^2 \lg n+n^2 \lg \frac{n}{2}+n^2 \lg \frac{n}{4}+...+n^2 \lg \frac{n}{2^k} \; =n^2 \lg (n * \frac{n}{2} * \frac{n}{4} * ... * \frac{n}{2^k})=n^2 \lg n \lg 2=O(n^2 \lg n)$

题目 **2** ：假设谷歌公司在过去 n 天中的股票价格记录在数组 $A[1..n]$ 中，我们希望从中找出两天的价格，其价格增幅最大，即找到 $A[i]$ 和 $A[j]$ $(i \leq j)$ 使得 $M = A[j] - A[i]$ 的值最大，请设计一个时间复杂度不超过 $O(n \lg n)$ 的分治算法

答：

***算法思路：

取 **m** 为整个数组的中位数，将数组一分为二，**S1** 和 **S2**，分别求出 **S1**、**S2** 中的最远点值，并与边界分割点的距离最比较，求出距离最大的两点。对于 **S1**、**S2**，分别使用同样的方法，直到最后子集中的元素为 **2.**

***时间复杂度分析：

每次分割，子问题的数量为 **2**，分割 **1** 次，需要比较的为两个子集的最大值和边界值，共三次. 因此得到表达式：$T(n)=2T(\frac{n}{2})+4$

由主定理得：$T(n)=O(n)$

---

*

算法实现题

题目 3 ：问题描述：在与联盟的战斗中连续失败后，帝国撤退到最后一个据点。根据其强大的防御系统，帝国击退了联盟攻击的六波浪潮。经过几个不眠之夜，联盟将军亚瑟注意到防御系统的唯一弱点就是能源供应。该系统由 N 个核电站供电，其中任何一个都会使系统失效。

这位将军很快就派 N 名特工进行突击，这些特工进入了据点。不幸的是，由于帝国空军的袭击，他们未能降落在预期位置。作为一名经验丰富的将军，亚瑟很快意识到他需要重新安排计划。他现在要知道的第一件事是哪个特工离任何一个核电站最近。你是否可以帮助将军计算特工与核电站之间的最小距离？

答：

***算法思路：

将问题抽象成平面分治最小点对问题，先以 x 为坐标排序，以中位数将点集分成两堆，递归求每堆的最近距离 d，然后对两堆之间横坐标为 x[mid] - d 和 x[mid] + d 之间的点按照 y 排序，求是否有比 d 小的距离，递归进行，直到求出最小距离。

***伪代码：

input S ：agent flag=1,station flag =0

function solve （S）：

begin

if |s|=2 且 flag 不一致

then d:=s 中这两点的距离

else if flag 一致

then d:=max

else if|S|=0

then d:=max

else

begin

1.m:=S 中各点 x 坐标值的中位数，构造 S1 和 S2，使 S1=p S|px m 和 S2=p S|px>m

2.d1=solve(S1),d2=solve(S2)

3.dm:=min(d1,d2)

4. 设 P1 是 S1 中距垂直分割线 l 的距离在 dm 之内的所有点组成的集合， P2 是 S2 中距分割线 l 的距离在 dm 之内所有点组成的集合。将 P1 和 P2 中的点依其 y 坐标值从小到大排序，并设 P1*和 P2*是相应的已排好序的点列;

5. 通过扫描 P1*以及对于 P1*中每个点检查 P2*中与其距离在 dm 之内的所有点 (最多 6 个) 可以完成合并。当 P1*中的扫描指针逐次向上移动 时，P2*中的扫描指针可在宽为 2dm 的一个区间内移动。设 dl 是按 这种扫描方式找到的点对间的最小距离;

6.d=min(dm,dl)

end
return d
end

```
#include <iostream>
#include <cstdio>
#include <cstring>
#include <algorithm>
#include <cmath>
#include<iomanip>
using namespace std;

typedef long long LL;

const LL INF = 1000000000000;
const int N = 100010;

struct Node {
  LL x, y;
  int id;
  Node(LL x = 0, LL y = 0, int id = 0) :x(x), y(y), id(id) {}
  const bool operator < (const Node A) const {
    return x == A.x ? y < A.y : x < A.x;
  }
}no[2 * N];

int n;

double dis(int a, int b) {
  return sqrt((double)((no[a].x - no[b].x) * (no[a].x - no[b].x) + (no[a].y
    - no[b].y) * (no[a].y - no[b].y)));
}

double solve(int l, int r) {
  if (l == r)return INF;
  int mid = (l + r) >> 1;
  double a = solve(l, mid);
  double b = solve(mid + 1, r);
  double d = min(a, b);
  for (int i = mid; i >= l; --i) {
    if (no[mid].x - no[i].x > d)break;
    for (int j = mid + 1; j <= r; ++j) {
      if (no[j].x - no[i].x > d)break;
      double tmp = dis(i, j);
```

```
            if (no[i].id != no[j].id && tmp < d)d = tmp;
        }
    }
    return d;
}

int main() {
    int t;
    cin >> t;
    while (t--) {
        cin >> n;
        for (int i = 0; i < n; ++i) {
            cin >> no[i].x >> no[i].y;
            no[i].id = 1;
        }
        for (int i = 0; i < n; ++i) {
            cin >> no[i+n].x >> no[i+n].y;
            no[i + n].id = 2;
        }
        sort(no, no + 2 * n);
        double ans = solve(0, 2 * n - 1);
        cout << setprecision(3) << fixed <<ans << endl;
    }
}
```