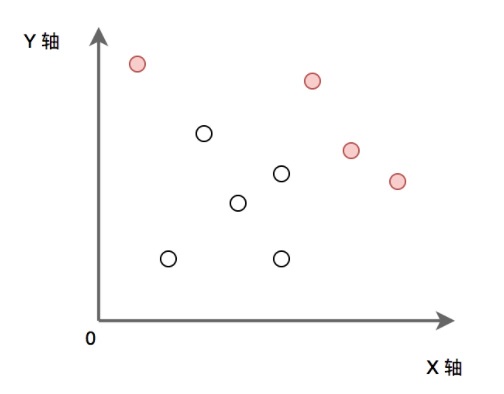
1、P为给定的二维平面整数点集。定义 P 中某点x，如果x满足 P 中任意点都不在 x 的右上方区域内（横纵坐标都大于x），则称其为“最大的”。求出所有“最大的”点的集合。（所有点的横坐标和纵坐标都不重复, 坐标轴范围在[0, 1e9) 内）

如下图：实心点为满足条件的点的集合。请实现代码找到集合 P 中的所有 ”最大“ 点的集合并输出。



**输入描述:**

第一行输入点集的个数 N， 接下来 N 行，每行两个数字代表点的 X 轴和 Y 轴。

对于 50%的数据, 1 <= N <= 10000;

对于 100%的数据, 1 <= N <= 500000;

**输出描述:**

输出“最大的” 点集合， 按照 X 轴从小到大的方式输出，每行两个数字分别代表点的 X 轴和 Y轴。

**输入例子1:**

5

1 2

5 3

4 6

7 5

9 0

**输出例子1:**

4 6

7 5

9 0

2、给定一个数组序列, 需要求选出一个区间, 使得该区间是所有区间中经过如下计算的值最大的一个：

区间中的最小数 \* 区间所有数的和最后程序输出经过计算后的最大值即可，不需要输出具体的区间。如给定序列  [6 2 1]则根据上述公式, 可得到所有可以选定各个区间的计算值:

[6] = 6 \* 6 = 36;

[2] = 2 \* 2 = 4;

[1] = 1 \* 1 = 1;

[6,2] = 2 \* 8 = 16;

[2,1] = 1 \* 3 = 3;

[6, 2, 1] = 1 \* 9 = 9;

从上述计算可见选定区间 [6] ，计算值为 36， 则程序输出为 36。

区间内的所有数字都在[0, 100]的范围内;

**输入描述:**

第一行输入数组序列长度n，第二行输入数组序列。

对于 50%的数据, 1 <= n <= 10000;

对于 100%的数据, 1 <= n <= 500000;

**输出描述:**

输出数组经过计算后的最大值。

**输入例子1:**

3

6 2 1

**输出例子1:**

36

3、给定一棵树的根节点, 在已知该树最大深度的情况下, 求节点数最多的那一层并返回具体的层数。

如果最后答案有多层, 输出最浅的那一层，树的深度不会超过100000。实现代码如下，请指出代码中的多处错误：

struct Node {

    vector<Node\*> sons;

};

void dfsFind(Node \*node, int dep, int counter[]) {

    counter[dep]++;

    for(int i = 0; i < node.sons.size(); i++) {

        dfsFind(node.sons[i], dep, counter);

    }

}

int find(Node \*root, int maxDep) {

    int depCounter[100000];

    dfsFind(root, 0, depCounter);

    int max, maxDep;

    for (int i = 1; i <= maxDep; i++) {

        if (depCounter[i] > max) {

            max = depCounter[i];

            maxDep = i;

        }

    }

    return maxDep;

}

4、某一个RPC服务A，对外提供接口MatchAds(AdTargetRequest req),发送请求，返回可展示的广告。如何测试这个服务接口的性能。

5、如果一个头条的客户端程序，冷启动时间为4秒，怎么判断开启速度是合理的还是不合理的？如果不合理，该如何找到问题，提供思路。