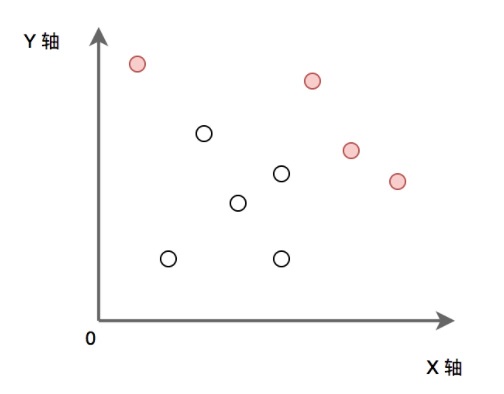
[编程题] 编程题1

时间限制：1秒

空间限制：32768K

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

例子说明1:

[编程题] 编程题2

时间限制：3秒

空间限制：131072K

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

例子说明1:

[编程题] 编程题3

时间限制：1秒

空间限制：81920K

产品经理(PM)有很多好的idea，而这些idea需要程序员实现。现在有N个PM，在某个时间会想出一个 idea，每个 idea 有提出时间、所需时间和优先等级。对于一个PM来说，最想实现的idea首先考虑优先等级高的，相同的情况下优先所需时间最小的，还相同的情况下选择最早想出的，没有 PM 会在同一时刻提出两个 idea。

同时有M个程序员，每个程序员空闲的时候就会查看每个PM尚未执行并且最想完成的一个idea,然后从中挑选出所需时间最小的一个idea独立实现，如果所需时间相同则选择PM序号最小的。直到完成了idea才会重复上述操作。如果有多个同时处于空闲状态的程序员，那么他们会依次进行查看idea的操作。

求每个idea实现的时间。

输入第一行三个数N、M、P，分别表示有N个PM，M个程序员，P个idea。随后有P行，每行有4个数字，分别是PM序号、提出时间、优先等级和所需时间。输出P行，分别表示每个idea实现的时间点。

输入描述:

输入第一行三个数N、M、P，分别表示有N个PM，M个程序员，P个idea。随后有P行，每行有4个数字，分别是PM序号、提出时间、优先等级和所需时间。全部数据范围 [1, 3000]。

输出描述:

输出P行，分别表示每个idea实现的时间点。

输入例子1:

2 2 5

1 1 1 2

1 2 1 1

1 3 2 2

2 1 1 2

2 3 5 5

输出例子1:

3

4

5

3

9

例子说明1:

[问答题]

题目描述

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;

}

[问答题]

题目描述

早期短链接广泛应用于图片上传网站，通过缩短网址URL链接字数，达到减少代码字符串的目的。常见于网店图片分类的使用，因有字符个数限制，采用短链接可以达到外链图片的目的。自微博盛行以来，在微博字数有限的特色下，短链接也盛行于微博网站，以节省字数给博主发布更多文字的空间。

问题描述：设计一个短链生成和查询系统，需要提供以下两个功能：

1、提供长链转换短链的接口

2、点击短链能跳转到对应的长链

题目要求：

1、同一个长链生成同一个短链接，不要有多个短链指向同一个长链。

2、同一个短链只能指向某一个长链，短链生成后要固定不变，不能再指向其它长链。

3、给出系统架构，需要考虑高并发解决方案。

4、考虑存储和缓存方案

数据量预估：

1、预计长链接总量500亿

2、长链换短链请求量：10W qps

3、短链跳转请求量：100W qps

1. #include<bits/stdc++.h>

#define frp(i,a,b) for(int i = a; i <= b; i++)

#define frpp(i,a,b) for(int i = a; i < b; i++)

#define frd(i,a,b) for(int i = a; i >= b; i--)

#define frdd(i,a,b) for(int i = a; i > b; i--)

#define clr(a,b) memset(a,b,sizeof(a))

#define pb push\_back

#define mp make\_pair

#define lb lower\_bound

#define ub upper\_bound

usingnamespacestd;

typedeflonglonglint;

structnod{

intx,y;

booloperator < (constnod & a)const{

returnx < a.x;

}

}a[500050];

intn;

inlinevoidget\_ready( )

{

scanf("%d", &n);

frp(i,1,n) scanf("%d%d",&a[i].x,&a[i].y);

}

inlinevoidsolve( )

{

sort(a+1,a+n+1);

stack<nod>qq;

intres;

frd(i,n,1){

if(qq.empty()){

qq.push(a[i]);

res = a[i].y;

}

else{

if(a[i].y>res){

qq.push(a[i]);

res = max(res,a[i].y);

}

}

}

while(!qq.empty()){

nod xx = qq.top( );

qq.pop( );

printf("%d %d\n",xx.x,xx.y);

}

}

intmain( )

{

get\_ready( );

solve( );

return0;

}