

Assignment: 编程作业—STL2

You have not submitted. You must earn 72/90 points to pass.

Deadline Pass this assignment by July 3, 11:59 PM PDT

Instructions (</learn/cpp-chengxu-sheji/programming/lorF0/bian-cheng-zuo-ye-stl2/submissions>) **Discussions** (</learn/cpp-chengxu-sheji/programming/lorF0/discussions>)

准备

在开始下面的作业前，请先点击[这里](https://d396qusza40orc.cloudfront.net/flex-spcpp/Programming%20Assignment%20PA6/PA6.zip)下载代码模版 (<https://d396qusza40orc.cloudfront.net/flex-spcpp/Programming%20Assignment%20PA6/PA6.zip>)。

编程题 # 1

来源: POJ (<http://cxsjsxmooc.openjudge.cn/test/9w6/>) (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)

注意： 总时间限制: **1000ms** 内存限制: **65536kB**

描述

下面的程序用枚举法解决如下问题，请填空。

平面上的一个矩形，如果其边平行于坐标轴，我们就称其为“标准矩形”。给定不重复的 n 个整点（横、纵坐标都是整数的点），求从这 n 个点中任取4点作为顶点所构成的四边形中，有多少个是标准矩形。

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
struct Point {
    int x;
    int y;
    Point(int x_,int y_):x(x_),y(y_) { }
};
bool operator < ( const Point & p1, const Point & p2)
{
    if( p1.y < p2.y )
        return true;
    else if( p1.y == p2.y )
        return p1.x < p2.x;
    else
        return false;
}
int main()
{
    int t;
    int x,y;
    cin >> t;
    vector<Point> v;
    while( t -- ) {
        cin >> x >> y;
        v.push_back(Point(x,y));
    }
    vector<Point>::iterator i,j;
    int nTotalNum = 0;
    // 在此处补充你的代码
    return 0;
}
```

输入

第一行是点的数目

其后每一行都代表一个点，由两个整数表示，第一个是x坐标，第二个是y坐标

输出

输出标准矩形的数目

样例输入

```
6
2 3
2 5
4 5
4 4
2 4
4 3
```

样例输出

```
3
```

提示

所缺代码具有如下形式：

```

_____;
for( i = v.begin(); i < v.end() - 1; i ++ )
    for(_____; _____; _____) {
        if(binary_search(v.begin(), v.end(), Point( j->x, i->y)) &&
            _____ &&
            _____ &&
            _____ )
            nTotalNum ++;
    }
cout << _____;
```

编程题 # 2

来源: POJ (<http://cxsjsxmooc.openjudge.cn/test/9w7>)(Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)

注意： 总时间限制: **1000ms** 内存限制: **65536kB**

描述

写一个自己的 CMyistream_iterator 模板，使之能和 istream_iterator 模板达到一样的效果，即：

输入：

```
79 90 20 hello me
```

输出:

79

79,90,20

hello,me

```
#include <iostream>
#include <string>
using namespace std;
// 在此处补充你的代码
int main()
{
    CMyistream_iterator<int> inputInt(cin);
    int n1,n2,n3;
    n1 = * inputInt; //读入 n1
    int tmp = * inputInt;
    cout << tmp << endl;
    inputInt ++;
    n2 = * inputInt; //读入 n2
    inputInt ++;
    n3 = * inputInt; //读入 n3
    cout << n1 << "," << n2<< "," << n3 << endl;
    CMyistream_iterator<string> inputStr(cin);
    string s1,s2;
    s1 = * inputStr;
    inputStr ++;
    s2 = * inputStr;
    cout << s1 << "," << s2 << endl;
    return 0;
}
```

输入

79 90 20 hello me

输出

79

79,90,20

hello,me

样例输入

```
79 90 20 hello me
```

样例输出

```
79
79,90,20
hello,me
```

提示

istream_iterator模版使用说明：

其构造函数执行过程中就会要求输入，然后每次执行++，则读取输入流中的下一个项目，执行*则返回上次从输入流中读取的项目。例如，下面程序运行时，就会等待用户输入数据，输入数据后程序才会结束：

```
#include <iostream>
#include <iterator>
using namespace std;
int main() {
    istream_iterator<int> inputInt(cin);
    return 0;
}
```

下面程序运行时，如果输入 12 34 程序输出结果是： 12,12

```
#include <iostream>
#include <iterator>
using namespace std;
int main()
{
    istream_iterator<int> inputInt(cin);
    cout << * inputInt << "," << * inputInt << endl;
    return 0;
}
```

下面程序运行时，如果输入 12 34 56程序输出结果是： 12,56

```
#include <iostream>
#include <iterator>
using namespace std;
int main()
{
    istream_iterator<int> inputInt(cin);
    cout << * inputInt << "," ;
    inputInt ++;
    inputInt ++;
    cout << * inputInt;
    return 0;
}
```

编程题 # 3: Set

来源: POJ (<http://cxsjsxmooc.openjudge.cn/test/T/>) (Coursera声明: 在POJ上完成的习题将不会计入Coursera的最后成绩。)

注意: 总时间限制: **5000ms** 内存限制: **10000kB**

描述

现有一整数集（允许有重复元素），初始为空。我们定义如下操作：

add x 把x加入集合

del x 把集合中所有与x相等的元素删除

ask x 对集合中元素x的情况询问

对每种操作，我们要求进行如下输出。

add 输出操作后集合中x的个数

del 输出操作前集合中x的个数

ask 先输出0或1表示x是否曾被加入集合（0表示不曾加入），再输出当前集合中x的个数，中间用空格隔开。

输入

第一行是一个整数n，表示命令数。0<=n<=100000。

后面n行命令，如Description中所述。

输出

共n行，每行按要求输出。

样例输入

```
7
add 1
add 1
ask 1
ask 2
del 2
del 1
ask 1
```

样例输出

```
1
2
1 2
0 0
0
2
1 0
```

提示

Please use STL's set and multiset to finish the task

编程题 # 4： 字符串操作

来源: POJ (<http://cxsjsxmooc.openjudge.cn/test/U/>) (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)

注意： 总时间限制: **1000ms** 内存限制: **65536kB**

描述

给定n个字符串（从1开始编号），每个字符串中的字符位置从0开始编号，长度为1-500，现有如下若干操作：

copy N X L：取出第N个字符串第X个字符开始的长度为L的字符串。

add S1 S2：判断S1，S2是否为0-99999之间的整数，若是则将其转化为整数做加法，若不是，则作字符串加法，返回的值为一字符串。

find S N: 在第N个字符串中从左开始找寻S字符串, 返回其第一次出现的位置, 若没有找到, 返回字符串的长度。

rfind S N: 在第N个字符串中从右开始找寻S字符串, 返回其第一次出现的位置, 若没有找到, 返回字符串的长度。

insert S N X: 在第N个字符串的第X个字符位置中插入S字符串。

reset S N: 将第N个字符串变为S。

print N: 打印输出第N个字符串。

printall: 打印输出所有字符串。

over: 结束操作。

其中N, X, L可由find与rfind操作表达式构成, S, S1, S2可由copy与add操作表达式构成。

输入

第一行为一个整数n (n在1-20之间)

接下来n行为n个字符串, 字符串不包含空格及操作命令等。

接下来若干行为一系列操作, 直到over结束。

输出

根据操作提示输出对应字符串。

样例输入

```
3
329strjvc
0padfk48
Ifjoqwoqejr
insert copy 1 find 2 1 2 2 2
print 2
reset add copy 1 find 3 1 3 copy 2 find 2 2 2 3
print 3
insert a 3 2
printall
over
```

样例输出


```
0p29adfk48
358
329strjvc
0p29adfk48
35a8
```

提示

推荐使用string类中的相关操作函数。

编程题 # 5: 热血格斗场

来源: POJ (<http://cxsjsxmooc.openjudge.cn/test/V/>) (Coursera声明: 在POJ上完成的习题将不会计入Coursera的最后成绩。)

注意: 总时间限制: **1000ms** 内存限制: **65536kB**

描述

为了迎接08年的奥运会, 让大家更加了解各种格斗运动, facer新开了一家热血格斗场。格斗场实行会员制, 但是新来的会员不需要交入会费, 而只要同一名老会员打一场表演赛, 证明自己的实力。

我们假设格斗的实力可以用一个正整数表示, 成为实力值。另外, 每个人都有唯一的id, 也是一个正整数。为了使得比赛更好看, 每一个新队员都会选择与他实力最为接近的人比赛, 即比赛双方的实力值之差的绝对值越小越好, 如果有两个人的实力值与他差别相同, 则他会选择比他弱的那个(显然, 虐人必被虐好)。

不幸的是, Facer一不小心把比赛记录弄丢了, 但是他还保留着会员的注册记录。现在请你帮facer恢复比赛纪录, 按照时间顺序依次输出每场比赛双方的id。

输入

第一行一个数 n ($0 < n \leq 100000$), 表示格斗场新来的会员数(不包括facer)。以后 n 行每一行两个数, 按照入会的时间给出会员的id和实力值。一开始, facer就算是会员, id为1, 实力值1000000000。输入保证两人的实力值不同。

输出

N 行, 每行两个数, 为每场比赛双方的id, 新手的id写在前面。

样例输入

```
3
2 1
3 3
4 2
```

样例输出

```
2 1
3 2
4 2
```

编程题 # 6: priority queue练习题

来源: POJ (<http://cxsjsxmooc.openjudge.cn/test/W/>) (Coursera声明: 在POJ上完成的习题将不会计入Coursera的最后成绩。)

注意: 总时间限制: 2500ms 内存限制: 131072kB

描述

我们定义一个正整数a比正整数b优先的含义是:

*a的质因数数目 (不包括自身) 比b的质因数数目多;

*当两者质因数数目相等时, 数值较大者优先级高。

现在给定一个容器, 初始元素数目为0, 之后每次往里面添加10个元素, 每次添加之后, 要求输出优先级最高与最低的元素, 并把该两元素从容器中删除。

输入

第一行: num (添加元素次数, num <= 30)

下面10*num行, 每行一个正整数n (n < 10000000)。

输出

每次输入10个整数后, 输出容器中优先级最高与最低的元素, 两者用空格间隔。

样例输入

```
1
10 7 66 4 5 30 91 100 8 9
```

样例输出

66 5

How to submit

When you're ready to submit, you can upload files for each part of the assignment on the "My submission" tab.

