# W1.A+B Problem

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/A/)(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

Calculate a + b

### 输入

Two integer a,,b (0 ≤ a,b ≤ 10)

### 输出

Output a + b

### 样例输入

### 1 2

### 样例输出

### 3

# W2.编程作业: 编程作业—C++初探

简单的学生信息处理程序实现

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/C/) (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

在一个学生信息处理程序中，要求实现一个代表学生的类，并且所有成员变量都应该是私有的。

（注：评测系统无法自动判断变量是否私有。我们会在结束之后统一对作业进行检查，请同学们严格按照题目要求完成，否则可能会影响作业成绩。）

输入

姓名，年龄，学号，第一学年平均成绩，第二学年平均成绩，第三学年平均成绩，第四学年平均成绩。

其中姓名、学号为字符串，不含空格和逗号；年龄为正整数；成绩为非负整数。

各部分内容之间均用单个英文逗号","隔开，无多余空格。

输出

一行，按顺序输出：姓名，年龄，学号，四年平均成绩（向下取整）。

各部分内容之间均用单个英文逗号","隔开，无多余空格。

样例输入

Tom,18,7817,80,80,90,70

### 样例输出

Tom,18,7817,80

# W3. 编程作业: 编程作业—类和对象

## 编程题＃1

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/3w7) (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

下面程序输出的结果是：

0

5

请填空：

#include <iostream>

using namespace std;

class A {

public:

int val;

// 在此处补充你的代码

};

main() {

A a;

cout << a.val << endl;

a.GetObj() = 5;

cout << a.val << endl;

}

### 输入

无

### 输出

0

5

### 样例输入

无

### 样例输出

0

5

### 提示

所缺代码具有如下形式：

A(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ){ val = n; }

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ GetObj() {

return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

}

## 编程题 ＃2

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/3w8/) (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

下面程序的输出是：

10

请补足Sample类的成员函数。不能增加成员变量。

#include <iostream>

using namespace std;

class Sample{

public:

int v;

Sample(int n):v(n) { }

// 在此处补充你的代码

};

int main() {

Sample a(5);

Sample b = a;

cout << b.v;

return 0;

}

### 输入

无

### 输出

10

### 样例输入

无

### 样例输出

10

## 编程题 ＃3

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/3w9/) (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

下面程序的输出结果是：

5,5

5,5

请填空：

#include <iostream>

using namespace std;

class Base {

public:

int k;

Base(int n):k(n) { }

};

class Big {

public:

int v; Base b;

// 在此处补充你的代码

};

int main() {

Big a1(5); Big a2 = a1;

cout << a1.v << "," << a1.b.k << endl;

cout << a2.v << "," << a2.b.k << endl;

return 0;

}

### 输入

无

### 输出

5,5

5,5

### 样例输入

无

### 样例输出

5,5

5,5

### 提示

所缺代码具有如下形式：

Big \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_{ }

Big \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_{ }

## 编程题＃4：魔兽世界之一：备战

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/B/)(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

魔兽世界的西面是红魔军的司令部，东面是蓝魔军的司令部。两个司令部之间是依次排列的若干城市。

红司令部，City 1，City 2，……，City n，蓝司令部

两军的司令部都会制造武士。武士一共有 dragon 、ninja、iceman、lion、wolf 五种。每种武士都有编号、生命值、攻击力这三种属性。

双方的武士编号都是从1开始计算。红方制造出来的第n个武士，编号就是n。同样，蓝方制造出来的第n个武士，编号也是n。

武士在刚降生的时候有一个生命值。

在每个整点，双方的司令部中各有一个武士降生。

红方司令部按照iceman、lion、wolf、ninja、dragon的顺序循环制造武士。

蓝方司令部按照lion、dragon、ninja、iceman、wolf的顺序循环制造武士。

制造武士需要生命元。

制造一个初始生命值为m的武士，司令部中的生命元就要减少m个。

如果司令部中的生命元不足以制造某个按顺序应该制造的武士，那么司令部就试图制造下一个。如果所有武士都不能制造了，则司令部停止制造武士。

给定一个时间，和双方司令部的初始生命元数目，要求你将从0点0分开始到双方司令部停止制造武士为止的所有事件按顺序输出。

一共有两种事件，其对应的输出样例如下：

1) 武士降生

输出样例： 004 blue lion 5 born with strength 5,2 lion in red headquarter

表示在4点整，编号为5的蓝魔lion武士降生，它降生时生命值为5，降生后蓝魔司令部里共有2个lion武士。（为简单起见，不考虑单词的复数形式）注意，每制造出一个新的武士，都要输出此时司令部里共有多少个该种武士。

2) 司令部停止制造武士

输出样例： 010 red headquarter stops making warriors

表示在10点整，红方司令部停止制造武士

输出事件时：

首先按时间顺序输出；

同一时间发生的事件，先输出红司令部的，再输出蓝司令部的。

### 输入

第一行是一个整数，代表测试数据组数。

每组测试数据共两行。

第一行：一个整数M。其含义为， 每个司令部一开始都有M个生命元( 1 <= M <= 10000)。

第二行：五个整数，依次是 dragon 、ninja、iceman、lion、wolf 的初始生命值。它们都大于0小于等于10000。

### 输出

对每组测试数据，要求输出从0时0分开始，到双方司令部都停止制造武士为止的所有事件。

对每组测试数据，首先输出"Case:n" n是测试数据的编号，从1开始 。

接下来按恰当的顺序和格式输出所有事件。每个事件都以事件发生的时间开头，时间以小时为单位，有三位。

### 样例输入

1

20

3 4 5 6 7

### 样例输出

Case:1

000 red iceman 1 born with strength 5,1 iceman in red headquarter

000 blue lion 1 born with strength 6,1 lion in blue headquarter

001 red lion 2 born with strength 6,1 lion in red headquarter

001 blue dragon 2 born with strength 3,1 dragon in blue headquarter

002 red wolf 3 born with strength 7,1 wolf in red headquarter

002 blue ninja 3 born with strength 4,1 ninja in blue headquarter

003 red headquarter stops making warriors

003 blue iceman 4 born with strength 5,1 iceman in blue headquarter

004 blue headquarter stops making warriors

# W4.编程作业: 编程作业—运算符重载

## 编程题 ＃1

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/4w3/) (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

下面程序的输出是：

3+4i

5+6i

请补足Complex类的成员函数。不能加成员变量。

#include <iostream>

#include <cstring>

#include <cstdlib>

using namespace std;

class Complex {

private:

double r,i;

public:

void Print() {

cout << r << "+" << i << "i" << endl;

}

// 在此处补充你的代码

};

int main() {

Complex a;

a = "3+4i"; a.Print();

a = "5+6i"; a.Print();

return 0;

}

### 输入

无

### 输出

3+4i

5+6i

## 编程题 ＃2

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/4w4) (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MyInt类只有一个成员变量。MyInt类内部的部分代码被隐藏了。假设下面的程序能编译通过，且输出结果是:

4,1

请写出被隐藏的部分。（您写的内容必须是能全部放进 MyInt类内部的，MyInt的成员函数里不允许使用静态变量）。

## 编程题 ＃3

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/4w5) (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

写一个二维数组类 Array2,使得下面程序的输出结果是：

0,1,2,3,

4,5,6,7,

8,9,10,11,

next

0,1,2,3,

4,5,6,7,

8,9,10,11,

程序：

#include <iostream>

#include <cstring>

using namespace std;

// 在此处补充你的代码

int main() {

Array2 a(3,4);

int i,j;

for( i = 0;i < 3; ++i )

for( j = 0; j < 4; j ++ )

a[i][j] = i \* 4 + j;

for( i = 0;i < 3; ++i ) {

for( j = 0; j < 4; j ++ ) {

cout << a(i,j) << ",";

}

cout << endl;

}

cout << "next" << endl;

Array2 b; b = a;

for( i = 0;i < 3; ++i ) {

for( j = 0; j < 4; j ++ ) {

cout << b[i][j] << ",";

}

cout << endl;

}

return 0;

}

输入

无

输出

0,1,2,3,

4,5,6,7,

8,9,10,11,

next

0,1,2,3,

4,5,6,7,

8,9,10,11,

## 编程题＃4：大整数的加减乘除

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/D/) (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

给出两个**正整数**以及四则运算操作符（+ - \* /），求运算结果。

### 输入

第一行：正整数a，长度不超过100

第二行：四则运算符o，o是“+”，“-”，“\*”，“/”中的某一个

第三行：正整数b，长度不超过100

保证输入不含多余的空格或其它字符

### 输出

一行：表达式“a o b”的值。

补充说明：

1. 减法结果有可能为负数

2. 除法结果向下取整

3. 输出符合日常书写习惯，不能有多余的0、空格或其它字符

### 样例输入

9876543210

+

9876543210

### 样例输出

19753086420

# W5. 编程作业: 编程作业—继承与派生

## 编程题＃1

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/5w5/) (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MyString 类，使得下面程序的输出结果是：

1. abcd-efgh-abcd-

2. abcd-

3.

4. abcd-efgh-

5. efgh-

6. c

7. abcd-

8. ijAl-

9. ijAl-mnop

10. qrst-abcd-

11. abcd-qrst-abcd- uvw xyz

about

big

me

take

abcd

qrst-abcd-

要求：MyString类必须是从C++的标准类string类派生而来。提示1：如果将程序中所有 "MyString" 用"string" 替换，那么题目的程序中除了最后两条语句编译无法通过外，其他语句都没有问题，而且输出和前面给的结果吻合。也就是说，MyString类对 string类的功能扩充只体现在最后两条语句上面。提示2: string类有一个成员函数 string substr(int start,int length); 能够求从 start位置开始，长度为length的子串

程序：

#include <iostream>

#include <cstring>

#include <cstdlib>

using namespace std;

// 在此处补充你的代码

int CompareString( const void \* e1, const void \* e2) {

MyString \* s1 = (MyString \* ) e1;

MyString \* s2 = (MyString \* ) e2;

if( \*s1 < \*s2 ) return -1;

else if( \*s1 == \*s2 ) return 0;

else if( \*s1 > \*s2 ) return 1;

}

int main() {

MyString s1("abcd-"),s2,s3("efgh-"),s4(s1);

MyString SArray[4] = {"big","me","about","take"};

cout << "1. " << s1 << s2 << s3<< s4<< endl;

s4 = s3; s3 = s1 + s3;

cout << "2. " << s1 << endl;

cout << "3. " << s2 << endl;

cout << "4. " << s3 << endl;

cout << "5. " << s4 << endl;

cout << "6. " << s1[2] << endl;

s2 = s1; s1 = "ijkl-";

s1[2] = 'A' ;

cout << "7. " << s2 << endl;

cout << "8. " << s1 << endl;

s1 += "mnop";

cout << "9. " << s1 << endl;

s4 = "qrst-" + s2;

cout << "10. " << s4 << endl;

s1 = s2 + s4 + " uvw " + "xyz";

cout << "11. " << s1 << endl;

qsort(SArray,4,sizeof(MyString), CompareString);

for( int i = 0;i < 4;++i )

cout << SArray[i] << endl;

//输出s1从下标0开始长度为4的子串

cout << s1(0,4) << endl;

//输出s1从下标为5开始长度为10的子串

cout << s1(5,10) << endl;

return 0;

}

### 输入

无

### 输出

1. abcd-efgh-abcd-

2. abcd-

3.

4. abcd-efgh-

5. efgh-

6. c

7. abcd-

8. ijAl-

9. ijAl-mnop

10. qrst-abcd-

11. abcd-qrst-abcd- uvw xyz

about

big

me

take

abcd

qrst-abcd-

## 编程题＃2： 魔兽世界之二：装备

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/E/) (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

魔兽世界的西面是红魔军的司令部，东面是蓝魔军的司令部。两个司令部之间是依次排列的若干城市。

红司令部，City 1，City 2，……，City n，蓝司令部

两军的司令部都会制造武士。武士一共有 dragon 、ninja、iceman、lion、wolf 五种。每种武士都有编号、生命值这两种属性。

有的武士可以拥有武器。武器有三种，sword, bomb,和arrow，编号分别为0,1,2。

双方的武士编号都是从1开始计算。红方制造出来的第 n 个武士，编号就是n。同样，蓝方制造出来的第 n 个武士，编号也是n。

不同的武士有不同的特点。

dragon 可以拥有一件武器。编号为n的dragon降生时即获得编号为 n%3 的武器。dragon还有“士气”这个属性，是个浮点数，其值为它降生后其司令部剩余生命元的数量除以造dragon所需的生命元数量。

ninjia可以拥有两件武器。编号为n的ninjia降生时即获得编号为 n%3 和 (n+1)%3的武器。

iceman有一件武器。编号为n的iceman降生时即获得编号为 n%3 的武器。

lion 有“忠诚度”这个属性，其值等于它降生后其司令部剩余生命元的数目。

wolf没特点。

请注意，在以后的题目里，武士的士气，生命值，忠诚度在其生存期间都可能发生变化，都有作用，武士手中的武器随着使用攻击力也会发生变化。

武士在刚降生的时候有一个生命值。

在每个整点，双方的司令部中各有一个武士降生。

红方司令部按照 iceman、lion、wolf、ninja、dragon 的顺序循环制造武士。

蓝方司令部按照 lion、dragon、ninja、iceman、wolf 的顺序循环制造武士。

制造武士需要生命元。

制造一个初始生命值为 m 的武士，司令部中的生命元就要减少 m 个。

如果司令部中的生命元不足以制造某个按顺序应该制造的武士，那么司令部就试图制造下一个。如果所有武士都不能制造了，则司令部停止制造武士。

给定一个时间，和双方司令部的初始生命元数目，要求你将从0点0分开始到双方司令部停止制造武士为止的所有事件按顺序输出。

一共有两种事件，其对应的输出样例如下：

1) 武士降生

输出样例： 004 blue lion 5 born with strength 5,2 lion in red headquarter

表示在 4点整，编号为5的蓝魔lion武士降生，它降生时生命值为5,降生后蓝魔司令部里共有2个lion武士。(为简单起见，不考虑单词的复数形式)注意，每制造出一个新的武士，都要输出此时司令部里共有多少个该种武士。

如果造出的是dragon，那么还要输出一行，例：

It has a arrow,and it's morale is 23.34

表示该dragon降生时得到了arrow,其士气是23.34（为简单起见，本题中arrow前面的冠词用a,不用an，士气精确到小数点后面2位，四舍五入）

如果造出的是ninjia，那么还要输出一行，例：

It has a bomb and a arrow

表示该ninjia降生时得到了bomb和arrow。

如果造出的是iceman，那么还要输出一行，例：

It has a sword

表示该iceman降生时得到了sword。

如果造出的是lion，那么还要输出一行，例：

It's loyalty is 24

表示该lion降生时的忠诚度是24。

2) 司令部停止制造武士

输出样例： 010 red headquarter stops making warriors

表示在 10点整，红方司令部停止制造武士

输出事件时：

首先按时间顺序输出；

同一时间发生的事件，先输出红司令部的，再输出蓝司令部的。

### 输入

第一行是一个整数,代表测试数据组数。

每组测试数据共两行。

第一行，一个整数M。其含义为： 每个司令部一开始都有M个生命元( 1 <= M <= 10000)

第二行：五个整数，依次是 dragon 、ninja、iceman、lion、wolf 的初始生命值。它们都大于0小于等于10000

### 输出

对每组测试数据，要求输出从0时0分开始，到双方司令部都停止制造武士为止的所有事件。

对每组测试数据，首先输出“Case:n" n是测试数据的编号，从1开始

接下来按恰当的顺序和格式输出所有事件。每个事件都以事件发生的时间开头，时间以小时为单位，有三位。

### 样例输入

1

20

3 4 5 6 7

### 样例输出

Case:1

000 red iceman 1 born with strength 5,1 iceman in red headquarter

It has a bomb

000 blue lion 1 born with strength 6,1 lion in blue headquarter

It's loyalty is 14

001 red lion 2 born with strength 6,1 lion in red headquarter

It's loyalty is 9

001 blue dragon 2 born with strength 3,1 dragon in blue headquarter

It has a arrow,and it's morale is 3.67

002 red wolf 3 born with strength 7,1 wolf in red headquarter

002 blue ninja 3 born with strength 4,1 ninja in blue headquarter

It has a sword and a bomb

003 red headquarter stops making warriors

003 blue iceman 4 born with strength 5,1 iceman in blue headquarter

It has a bomb

004 blue headquarter stops making warriors

# W6.编程作业: 编程作业—多态与虚函数

## 编程题 #1

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/6w4/) (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**

下面程序的输出结果是：

**A::Fun**

**C::Do**

请填空：

#include <iostream>

using namespace std;

class A {

private:

int nVal;

public:

void Fun()

{ cout << "A::Fun" << endl; }

void Do()

{ cout << "A::Do" << endl; }

};

class B:public A {

public:

virtual void Do()

{ cout << "B::Do" << endl; }

};

class C:public B {

public:

void Do( )

{ cout << "C::Do" <<endl; }

void Fun()

{ cout << "C::Fun" << endl; }

};

void Call(

// 在此处补充你的代码

) {

p.Fun(); p.Do();

}

int main() {

C c; Call(c);

return 0;

}

输入

无

输出

**A::Fun**

**C::Do**

## 编程题＃2

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/6w5/) (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

下面程序的输出结果是：

destructor B

destructor A

请完整写出 class A。 限制条件：不得为 class A 编写构造函数。

#include <iostream>

using namespace std;

class A {

// 在此处补充你的代码

};

class B:public A {

public:

~B() { cout << "destructor B" << endl; }

};

int main() {

A \* pa;

pa = new B;

delete pa;

return 0;

}

输入

无

输出

destructor B

destructor A

## 编程题 #3

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/6w6) (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

下面的程序输出结果是：

A::Fun

A::Do

A::Fun

C::Do

请填空：

#include <iostream>

using namespace std;

class A {

private:

int nVal;

public:

void Fun()

{ cout << "A::Fun" << endl; }

virtual void Do()

{ cout << "A::Do" << endl; }

};

class B:public A {

public:

virtual void Do()

{ cout << "B::Do" << endl; }

};

class C:public B {

public:

void Do( )

{ cout << "C::Do" << endl; }

void Fun()

{ cout << "C::Fun" << endl; }

};

void Call(

// 在此处补充你的代码

) {

p->Fun(); p->Do();

}

int main() {

Call( new A() );

Call( new C() );

return 0;

}

输入

无

输出

A::Fun

A::Do

A::Fun

C::Do

## 编程题＃4： 魔兽世界终极版

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/R/) (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: 2000ms 内存限制: 65536kB**

描述

魔兽世界的西面是红魔军的司令部，东面是蓝魔军的司令部。两个司令部之间是依次排列的若干城市，城市从西向东依次编号为1,2,3 .... N ( N <= 20 )。红魔军的司令部算作编号为0的城市，蓝魔军的司令部算作编号为N+1的城市。司令部有生命元，用于制造武士。

两军的司令部都会制造武士。武士一共有 dragon 、ninja、iceman、lion、wolf 五种。每种武士都有编号、生命值、攻击力这三种属性。

双方的武士编号都是从1开始计算。红方制造出来的第 n 个武士，编号就是n。同样，蓝方制造出来的第 n 个武士，编号也是n。

武士在刚降生的时候有一个初始的生命值，生命值在战斗中会发生变化，如果生命值减少到0（生命值变为负数时应当做变为0处理），则武士死亡（消失）。

有的武士可以拥有武器。武器有三种，sword, bomb,和arrow，编号分别为0,1,2。

武士降生后就朝对方司令部走，在经过的城市如果遇到敌人（同一时刻每个城市最多只可能有1个蓝武士和一个红武士），就会发生战斗。每次战斗只有一方发起主动进攻一次。被攻击者生命值会减去进攻者的攻击力值和进攻者手中sword的攻击力值。被进攻者若没死，就会发起反击，被反击者的生命值要减去反击者攻击力值的一半(去尾取整)和反击者手中sword的攻击力值。反击可能致敌人于死地。

如果武士在战斗中杀死敌人（不论是主动进攻杀死还是反击杀死），则其司令部会立即向其发送8个生命元作为奖励，使其生命值增加8。当然前提是司令部得有8个生命元。如果司令部的生命元不足以奖励所有的武士，则优先奖励距离敌方司令部近的武士。

如果某武士在某城市的战斗中杀死了敌人，则该武士的司令部立即取得该城市中所有的生命元。注意，司令部总是先完成全部奖励工作，然后才开始从各个打了胜仗的城市回收生命元。对于因司令部生命元不足而领不到奖励的武士，司令部也不会在取得战利品生命元后为其补发奖励。

如果一次战斗的结果是双方都幸存(平局)，则双方都不能拿走发生战斗的城市的生命元。

城市可以插旗子，一开始所有城市都没有旗子。在插红旗的城市，以及编号为奇数的无旗城市，由红武士主动发起进攻。在插蓝旗的城市，以及编号为偶数的无旗城市，由蓝武士主动发起进攻。

当某个城市有连续两场战斗都是同一方的武士杀死敌人(两场战斗之间如果有若干个战斗时刻并没有发生战斗，则这两场战斗仍然算是连续的；但如果中间有平局的战斗，就不算连续了) ，那么该城市就会插上胜方的旗帜，若原来插着败方的旗帜，则败方旗帜落下。旗帜一旦插上，就一直插着，直到被敌人更换。一个城市最多只能插一面旗帜，旗帜没被敌人更换前，也不会再次插同颜色的旗。

各种武器有其特点：

sword武器的初始攻击力为拥有它的武士的攻击力的20%（去尾取整）。但是sword每经过一次战斗(不论是主动攻击还是反击)，就会变钝，攻击力变为本次战斗前的80% (去尾取整)。sword攻击力变为0时，视为武士失去了sword。如果武士降生时得到了一个初始攻击力为0的sword，则视为武士没有sword.

arrow有一个攻击力值R。如果下一步要走到的城市有敌人，那么拥有arrow的武士就会放箭攻击下一个城市的敌人（不能攻击对方司令部里的敌人）而不被还击。arrow使敌人的生命值减少R，若减至小于等于0，则敌人被杀死。arrow使用3次后即被耗尽，武士失去arrow。两个相邻的武士可能同时放箭把对方射死。

拥有bomb的武士，在战斗开始前如果判断自己将被杀死（不论主动攻击敌人，或者被敌人主动攻击都可能导致自己被杀死，而且假设武士可以知道敌人的攻击力和生命值），那么就会使用bomb和敌人同归于尽。武士不预测对方是否会使用bomb。

武士使用bomb和敌人同归于尽的情况下，不算是一场战斗，双方都不能拿走城市的生命元，也不影响城市的旗帜。

不同的武士有不同的特点。

dragon可以拥有一件武器。编号为n的dragon降生时即获得编号为 n%3 的武器。dragon还有“士气”这个属性，是个浮点数，其值为它降生后其司令部剩余生命元的数量除以造dragon所需的生命元数量。dragon 在一次在它主动进攻的战斗结束后，如果还没有战死，而且士气值大于0.8，就会欢呼。dragon每取得一次战斗的胜利(敌人被杀死)，士气就会增加0.2，每经历一次未能获胜的战斗，士气值就会减少0.2。士气增减发生在欢呼之前。

ninjia可以拥有两件武器。编号为n的ninjia降生时即获得编号为 n%3 和 (n+1)%3的武器。ninja 挨打了也从不反击敌人。

iceman有一件武器。编号为n的iceman降生时即获得编号为 n%3 的武器。iceman 每前进两步，在第2步完成的时候，生命值会减少9，攻击力会增加20。但是若生命值减9后会小于等于0，则生命值不减9,而是变为1。即iceman不会因走多了而死。

lion 有“忠诚度”这个属性，其初始值等于它降生之后其司令部剩余生命元的数目。每经过一场未能杀死敌人的战斗，忠诚度就降低K。忠诚度降至0或0以下，则该lion逃离战场,永远消失。但是已经到达敌人司令部的lion不会逃跑。Lion在己方司令部可能逃跑。lion 若是战死，则其战斗前的生命值就会转移到对手身上。所谓“战斗前”，就是每个小时的40分前的一瞬间。

wolf降生时没有武器，但是在战斗中如果获胜（杀死敌人），就会缴获敌人的武器，但自己已有的武器就不缴获了。被缴获的武器当然不能算新的，已经被用到什么样了，就是什么样的。

以下是不同时间会发生的不同事件：

在每个整点，即每个小时的第0分， 双方的司令部中各有一个武士降生。

红方司令部按照 iceman、lion、wolf、ninja、dragon 的顺序制造武士。

蓝方司令部按照 lion、dragon、ninja、iceman、wolf 的顺序制造武士。

制造武士需要生命元。

制造一个初始生命值为 m 的武士，司令部中的生命元就要减少 m 个。

如果司令部中的生命元不足以制造某武士，那么司令部就等待，直到获得足够生命元后的第一个整点，才制造该武士。例如，在2:00，红方司令部本该制造一个 wolf ，如果此时生命元不足，那么就会等待，直到生命元足够后的下一个整点，才制造一个 wolf。

在每个小时的第5分，该逃跑的lion就在这一时刻逃跑了。

在每个小时的第10分：所有的武士朝敌人司令部方向前进一步。即从己方司令部走到相邻城市，或从一个城市走到下一个城市。或从和敌军司令部相邻的城市到达敌军司令部。

在每个小时的第20分：每个城市产出10个生命元。生命元留在城市，直到被武士取走。

在每个小时的第30分：如果某个城市中只有一个武士，那么该武士取走该城市中的所有生命元，并立即将这些生命元传送到其所属的司令部。

在每个小时的第35分，拥有arrow的武士放箭，对敌人造成伤害。放箭事件应算发生在箭发出的城市。注意，放箭不算是战斗，因此放箭的武士不会得到任何好处。武士在没有敌人的城市被箭射死也不影响其所在城市的旗帜更换情况。

在每个小时的第38分，拥有bomb的武士评估是否应该使用bomb。如果是，就用bomb和敌人同归于尽。

在每个小时的第40分：在有两个武士的城市，会发生战斗。 如果敌人在5分钟前已经被飞来的arrow射死，那么仍然视为发生了一场战斗，而且存活者视为获得了战斗的胜利。此情况下不会有“武士主动攻击”，“武士反击”，“武士战死”的事件发生，但战斗胜利后应该发生的事情都会发生。如Wolf一样能缴获武器，旗帜也可能更换，等等。在此情况下,Dragon同样会通过判断是否应该轮到自己主动攻击来决定是否欢呼。

在每个小时的第50分，司令部报告它拥有的生命元数量。

在每个小时的第55分，每个武士报告其拥有的武器情况。

武士到达对方司令部后就算完成任务了，从此就呆在那里无所事事。

任何一方的司令部里若是出现了2个敌人，则认为该司令部已被敌人占领。

任何一方的司令部被敌人占领，则战争结束。战争结束之后就不会发生任何事情了。

给定一个时间，要求你将从0点0分开始到此时间为止的所有事件按顺序输出。事件及其对应的输出样例如下：

**1) 武士降生**

输出样例： 000:00 blue lion 1 born

表示在 0点0分，编号为1的蓝魔lion武士降生

如果造出的是dragon，那么还要多输出一行，例：

000:00 blue dragon 1 born

Its morale is 23.34

表示该该dragon降生时士气是23. 34(四舍五入到小数点后两位)

如果造出的是lion，那么还要多输出一行，例:

000:00 blue lion 1 born

Its loyalty is 24

表示该lion降生时的忠诚度是24

**2) lion逃跑**

输出样例： 000:05 blue lion 1 ran away

表示在 0点5分，编号为1的蓝魔lion武士逃走

**3) 武士前进到某一城市**

输出样例： 000:10 red iceman 1 marched to city 1 with 20 elements and force 30

表示在 0点10分，红魔1号武士iceman前进到1号城市，此时他生命值为20,攻击力为30

对于iceman,输出的生命值和攻击力应该是变化后的数值

**4)武士放箭**

输出样例： 000:35 blue dragon 1 shot

表示在 0点35分，编号为1的蓝魔dragon武士射出一支箭。如果射出的箭杀死了敌人，则应如下输出：

000:35 blue dragon 1 shot and killed red lion 4

表示在 0点35分，编号为1的蓝魔dragon武士射出一支箭，杀死了编号为4的红魔lion。

**5)武士使用bomb**

输出样例： 000:38 blue dragon 1 used a bomb and killed red lion 7

表示在 0点38分，编号为1的蓝魔dragon武士用炸弹和编号为7的红魔lion同归于尽。

**6) 武士主动进攻**

输出样例：000:40 red iceman 1 attacked blue lion 1 in city 1 with 20 elements and force 30

表示在0点40分，1号城市中，红魔1号武士iceman 进攻蓝魔1号武士lion,在发起进攻前，红魔1号武士iceman生命值为20，攻击力为 30

**7) 武士反击**

输出样例：001:40 blue dragon 2 fought back against red lion 2 in city 1

表示在1点40分，1号城市中，蓝魔2号武士dragon反击红魔2号武士lion

**8) 武士战死**

输出样例：001:40 red lion 2 was killed in city 1

被箭射死的武士就不会有这一条输出。

**9) 武士欢呼**

输出样例：003:40 blue dragon 2 yelled in city 4

**10) 武士获取生命元( elements )**

输出样例：001:40 blue dragon 2 earned 10 elements for his headquarter

**11) 旗帜升起**

输出样例：004:40 blue flag raised in city 4

**12) 武士抵达敌军司令部**

输出样例：001:10 red iceman 1 reached blue headquarter with 20 elements and force 30

(此时他生命值为20,攻击力为30）对于iceman,输出的生命值和攻击力应该是变化后的数值

**13) 司令部被占领**

输出样例：003:10 blue headquarter was taken

**14)司令部报告生命元数量**

000:50 100 elements in red headquarter

000:50 120 elements in blue headquarter

表示在0点50分，红方司令部有100个生命元，蓝方有120个

**15)武士报告武器情况**

000:55 blue wolf 2 has arrow(2),bomb,sword(23)

000:55 blue wolf 4 has no weapon

000:55 blue wolf 5 has sword(20)

表示在0点55分，蓝魔2号武士wolf有一支arrow（这支arrow还可以用2次），一个bomb，还有一支攻击力为23的sword。

蓝魔4号武士wolf没武器。

蓝魔5号武士wolf有一支攻击力为20的sword。

交代武器情况时，次序依次是：arrow,bomb,sword。如果没有某种武器，某种武器就不用提。报告时，先按从西向东的顺序所有的红武士报告，然后再从西向东所有的蓝武士报告。

输出事件时：

首先按时间顺序输出；

同一时间发生的事件，按发生地点从西向东依次输出. 武士前进的事件, 算是发生在目的地。

在一次战斗中有可能发生上面的 6 至 11 号事件。这些事件都算同时发生，其时间就是战斗开始时间。一次战斗中的这些事件，序号小的应该先输出。

两个武士同时抵达同一城市，则先输出红武士的前进事件，后输出蓝武士的。

显然，13号事件发生之前的一瞬间一定发生了12号事件。输出时，这两件事算同一时间发生，但是应先输出12号事件

虽然任何一方的司令部被占领之后，就不会有任何事情发生了。但和司令部被占领同时发生的事件，全都要输出。

输入

第一行是t,代表测试数据组数

每组样例共三行。

第一行，五个整数 M,N,R,K, T。其含义为：

每个司令部一开始都有M个生命元( 1 <= M <= 10000)

两个司令部之间一共有N个城市( 1 <= N <= 20 )

arrow的攻击力是R

lion每经过一场未能杀死敌人的战斗，忠诚度就降低K。

要求输出从0时0分开始，到时间T为止(包括T) 的所有事件。T以分钟为单位，0 <= T <= 5000

第二行：五个整数，依次是 dragon 、ninja、iceman、lion、wolf 的初始生命值。它们都大于0小于等于10000

第三行：五个整数，依次是 dragon 、ninja、iceman、lion、wolf 的攻击力。它们都大于0小于等于10000

输出

对每组数据，先输出一行：

Case n:

如对第一组数据就输出 Case1:

然后按恰当的顺序和格式输出到时间T为止发生的所有事件。每个事件都以事件发生的时间开头，时间格式是“时: 分”，“时”有三位，“分”有两位。

样例输入

1

20 1 10 10 1000

20 20 30 10 20

5 5 5 5 5

样例输出

Case 1:

000:00 blue lion 1 born

Its loyalty is 10

000:10 blue lion 1 marched to city 1 with 10 elements and force 5

000:30 blue lion 1 earned 10 elements for his headquarter

000:50 20 elements in red headquarter

000:50 20 elements in blue headquarter

000:55 blue lion 1 has no weapon

001:00 blue dragon 2 born

Its morale is 0.00001:10 blue lion 1 reached red headquarter with 10 elements and force 5

001:10 blue dragon 2 marched to city 1 with 20 elements and force 5

001:30 blue dragon 2 earned 10 elements for his headquarter

001:50 20 elements in red headquarter

001:50 10 elements in blue headquarter

001:55 blue lion 1 has no weapon

001:55 blue dragon 2 has arrow(3)

002:10 blue dragon 2 reached red headquarter with 20 elements and force 5

002:10 red headquarter was taken

# W7.编程作业

## 编程题＃1

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/7w6/) (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

实现一个三维数组模版CArray3D，可以用来生成元素为任意类型变量的三维数组，使得下面程序输出结果是：

0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,

注意，只能写一个类模版，不能写多个。

#include <iostream>

using namespace std;

// 在此处补充你的代码

int main()

{

CArray3D<int> a(3,4,5);

int No = 0;

for( int i = 0; i < 3; ++ i )

for( int j = 0; j < 4; ++j )

for( int k = 0; k < 5; ++k )

a[i][j][k] = No ++;

for( int i = 0; i < 3; ++ i )

for( int j = 0; j < 4; ++j )

for( int k = 0; k < 5; ++k )

cout << a[i][j][k] << ",";

return 0;

}

输入

无

输出

0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,

样例输入

无

样例输出

0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29

    ,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54

    ,55,56,57,58,59,

提示

提示：类里面可以定义类，类模版里面也可以定义类模版。例如：

class A

{

class B {

};

};

template

class S

{

T x;

class K {

T a;

};

};

## 编程题＃2： 实数的输出格式

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/F/) (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: 1000kB**

描述

利用流操纵算子实现： 输入一个实数，先以非科学计数法输出，小数点后面保留5位有效数字；再以科学计数法输出，小数点后面保留7位有效数字。

**注意：在不同系统、编译器上的输出格式略有不同，但保证在程序中采用默认格式设置一定能在OJ平台上得到正确结果。**

输入

以非科学计数法表示的一个正实数，保证可以用double类型存储。

输出

第一行：以非科学计数法输出该实数，小数点后面保留5位有效数字；

第二行：以科学计数法输出该实数，小数点后面保留7位有效数字。

样例输入

12.34

样例输出

12.34000

1.2340000e+01

## 编程题＃3： 整数的输出格式

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/G/) (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: 1000kB**

描述

利用流操纵算子实现： 输入一个整数，先将该整数以十六进制输出，然后再将该整数以10个字符的宽度输出，宽度不足时在左边补0。

注意：在不同系统、编译器上的输出格式略有不同，但保证在程序中采用默认格式设置一定能在OJ平台上得到正确结果。

输入

一个正整数，保证可以用int类型存储。

输出

第一行：以十六进制输出该整数；

第二行：以10个字符的宽度输出该整数。

样例输入

23

样例输出

17

0000000023

## 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

Opadfk48

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

Op29adfk48

358

329strjvc

Op29adfk48

35a8

提示

推荐使用string类中的相关操作函数。

# W8.编程作业

## 程序填空题＃1

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/9w5/) (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**

下面的程序输出结果是:

1 2 6 7 8 9

请填空：

#include <iostream>

#include <iterator>

#include <set>

using namespace std;

int main() {

int a[] = {8,7,8,9,6,2,1};

// 在此处补充你的代码

ostream\_iterator<int> o(cout," ");

copy( v.begin(),v.end(),o);

return 0;

}

**输入**

无

**输出**

1 2 6 7 8 9

**样例输入**

无

**样例输出**

1 2 6 7 8 9

## 编程题＃2 List

[来源: POJ](http://cxsjsxmooc.openjudge.cn/test/S/) (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: 4000ms 内存限制: 65536kB**

### 描述

写一个程序完成以下命令：

new id ——新建一个指定编号为id的序列(id<10000)

add id num——向编号为id的序列加入整数num

merge id1 id2——合并序列id1和id2中的数，并将id2清空

unique id——去掉序列id中重复的元素

out id ——从小到大输出编号为id的序列中的元素，以空格隔开

### 输入

第一行一个数n，表示有多少个命令( n＜＝２０００００)。以后n行每行一个命令。

### 输出

按题目要求输出。

**样例输入**

16

new 1

new 2

add 1 1

add 1 2

add 1 3

add 2 1

add 2 2

add 2 3

add 2 4

out 1

out 2

merge 1 2

out 1

out 2

unique 1

out 1

**样例输出**

1 2 3

1 2 3 4

1 1 2 2 3 3 4

1 2 3 4

# W9.编程作业

## 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。)

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。)

总时间限制:

5000ms

内存限制:

100000kB

**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，中间用空格格开。

**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。)

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

Opadfk48

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**

Op29adfk48

358

329strjvc

Op29adfk48

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。)

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，每个人都有一个唯一的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 <=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。)

**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

# 期末考试

## 编程题＃1：输出200

**来源: 北京大学在线程序评测系统POJ (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)**

**总时间限制: 1000ms 内存限制: 1024kB**

### 描述

使以下代码输出结果为200.

#include<iostream>

using namespace std;

class Number {

public:

int num;

Number(int n=0): num(n) {}

// 在此处补充你的代码

};

int main() {

Number n1(10), n2(20);

Number n3;n3 = n1\*n2;

cout << int(n3) << endl;

return 0;

}

### 输入

不需要输入。

### 输出

输出结果为200。

### 样例输入

不需要输入。

### 样例输出

200

## 编程题＃2：输出指定结果一

**来源: 北京大学在线程序评测系统POJ (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)**

**总时间限制: 1000ms 内存限制: 1024kB**

### 描述

填写代码，使输出结果为

2

2

8

10

#include <iostream>

using namespace std;

class Number {

public:

int num;

Number(int n): num(n) {

}

// 在此处补充你的代码

};

int main() {

Number a(2);

Number b = a;

cout << a.value() << endl;

cout << b.value() << endl;

a.value() = 8;

cout << a.value() << endl;

a+b;

cout << a.value() << endl;

return 0;

}

### 输入

不需要输入。

### 输出

使输出结果为

2

2

8

10

## 编程题＃3：计算数列平方和

**来源: 北京大学在线程序评测系统POJ (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)**

**总时间限制: 1000ms 内存限制: 1024kB**

### 描述

请写出sum函数，使其可以计算输入数列的平方和。

#include <iostream>

using namespace std;

// 在此处补充你的代码

int sqr(int n) {

return n \* n;

}

int main() {

int t, n, a[0x100];

cin >> t;

for (int c = 0; c < t; ++c) {

cin >> n;

for (int i = 0; i < n; ++i) cin >> a[i];

cout << sum(a, n, sqr) << endl;

}

return 0;

}

### 输入

第一行是一个整数 t (t <= 10)，表示数据组数；

每组输入数据包含两行，第一行是一个整数 n (n <= 100)，

第二行是 n 个用空格分隔开的整数

### 输出

对每组输入数据，输出该组数据中 n 个整数的平方和

### 样例输入

2

2

4 3

3

0 1 2

### 样例输出

25

5

## 编程题＃4：计算整数平方和

**来源: 北京大学在线程序评测系统POJ (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)**

**总时间限制: 1000ms 内存限制: 1024kB**

### 描述

下列程序每次读入一个整数N，若N为0则退出，否则输出N和N的平方。

#include <iostream>

using namespace std;

// 在此处补充你的代码

int main(int argc, char\* argv[]) {

CType obj;

int n;

cin>>n;

while ( n ) {

obj.setvalue(n);

cout<<obj++<<" "<<obj<<endl;

cin>>n;

}

return 0;

}

### 输入

K个整数。除最后一个数据外，其他数据均不为0。

### 输出

K-1行。第I行输出第I个输入数和它的平方。

### 样例输入

1 5 8 9 0

### 样例输出

1 1

5 25

8 64

9 81

## 编程题＃5：计算数组的低3位之和

**来源: 北京大学在线程序评测系统POJ (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)**

**总时间限制: 1000ms 内存限制: 1024kB**

### 描述

输入一个正整数构成的数组a[0], a[1], a[2], ... , a[n-1], 计算它们的二进制低3位之和。

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

// 在此处补充你的代码

int main(int argc, char\* argv[]) {

int v, my\_sum=0;

vector<int> vec;

cin>>v;

while ( v ) {

vec.push\_back(v);

cin>>v;

}

for\_each(vec.begin(), vec.end(), CMy\_add(my\_sum));

cout<<my\_sum<<endl;

return 0;

}

### 输入

数组a,以0表示输入结束。

### 输出

一个整数 , 所输入数组各元素的二进制低3位之和。

### 样例输入

1 3 9 7 3 6 20 15 18 17 4 8 18 0

### 样例输出

41

## 编程题＃6：MyString

**来源: 北京大学在线程序评测系统POJ (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)**

**总时间限制: 1000ms 内存限制: 1024kB**

### 描述

请写出 MyString类，使得下面程序的输出结果符合下面的要求。

#include <cstring>

#include <cstdlib>

#include <string>

#include <iostream>

using namespace std;

// 在此处补充你的代码

int CompareString( const void \* e1, const void \* e2)

{

MyString \* s1 = (MyString \* ) e1;

MyString \* s2 = (MyString \* ) e2;

if( \* s1 < \*s2 )

return -1;

else if( \*s1 == \*s2)

return 0;

else if( \*s1 > \*s2 )

return 1;

}

int main()

{

MyString s1("abcd-"),s2,s3("efgh-"),s4(s1);

MyString SArray[4] = {"big","me","about","take"};

cout << "1. " << s1 << s2 << s3<< s4<< endl;

s4 = s3;

s3 = s1 + s3;

cout << "2. " << s1 << endl;

cout << "3. " << s2 << endl;

cout << "4. " << s3 << endl;

cout << "5. " << s4 << endl;

cout << "6. " << s1[2] << endl;

s2 = s1;

s1 = "ijkl-";

s1[2] = 'A' ;

cout << "7. " << s2 << endl;

cout << "8. " << s1 << endl;

s1 += "mnop";

cout << "9. " << s1 << endl;

s4 = "qrst-" + s2;

cout << "10. " << s4 << endl;

s1 = s2 + s4 + " uvw " + "xyz";

cout << "11. " << s1 << endl;

qsort(SArray,4,sizeof(MyString),CompareString);

for( int i = 0;i < 4;i ++ )

cout << SArray[i] << endl;

//s1的从下标0开始长度为4的子串

cout << s1(0,4) << endl;

//s1的从下标5开始长度为10的子串

cout << s1(5,10) << endl;

return 0;

}

### 输入

无

### 输出

1. abcd-efgh-abcd-

2. abcd-

3.

4. abcd-efgh-

5. efgh-

6. c

7. abcd-

8. ijAl-

9. ijAl-mnop

10. qrst-abcd-

11. abcd-qrst-abcd- uvw xyz

about

big

me

take

abcd

qrst-abcd-

## 编程题＃7：字符串排序

**来源: 北京大学在线程序评测系统POJ (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)**

**总时间限制: 1000ms 内存限制: 1024kB**

### 描述

请按照要求对输入的字符串进行排序。

#include <iostream>

#include <string>

#include <list>

using namespace std;

class A{

private:

string name;

public:

A(string n) :name(n){}

friend bool operator < (const class A& a1, const class A &a2);

friend bool operator == (const class A &a1, const class A &a2){

if (a1.name.size() == a2.name.size())

return true;

else

return false;

}

friend ostream & operator << (ostream &o, const A &a){

o << a.name;

return o;

}

string get\_name() const{

return name;

}

int get\_size() const{

return name.size();

}

};

// 在此处补充你的代码

int main(int argc, char\* argv[])

{

list<A> lst;

int ncase, n, i = 1;

string s;

cin >> ncase;

while (ncase--){

cout << "Case: "<<i++ << endl;

cin >> n;

for (int i = 0; i < n; i++){

cin >> s;

lst.push\_back(A(s));

}

lst.sort();

Show(lst.begin(), lst.end(), Print());

cout << endl;

lst.sort(MyLarge<A>());

Show(lst.begin(), lst.end(), Print());

cout << endl;

lst.clear();

}

return 0;

}

### 输入

第一行是正整数T，表示测试数据的组数

每组测试数据输入共两行，

第一行是正整数N，表示字符串个数

第二行是N个字符串, 字符串间用空格分离

### 输出

对于每组测试数据，先输出一行：

Case: n

如对第一组数据就输出Case: 1

第二行按照字符串长度从小到大排序之后输出N个字符串，字符串之间以空格间隔（不会出现字符串长度相同的情况）

第三行按照字符串首字符ASCII码序从小到大排序之后输出N个字符串，字符串之间以空格间隔（不会出现字符串首字母相同的情况）

### 样例输入

2

4

a bnss ds tsdfasg

5

aaa bbbb ccccd sa q

### 样例输出

Case: 1

a ds bnss tsdfasg

a bnss ds tsdfasg

Case: 2

q sa aaa bbbb ccccd

aaa bbbb ccccd q sa

## 编程题＃8：计算整数k

**来源: 北京大学在线程序评测系统POJ (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)**

**总时间限制: 1000ms 内存限制: 1024kB**

### 描述

输入整数 n ( 0 <=n<= 2^30) , 以及整数i,j(0 <= i,j <31,i < j-1), 输出整数k(按十六进制输出结果 )，k的第i位和n相同，第j位和n不同，i,j之间的位是1, 其他位都是0。这里提到的所有的位，指的都是二进制位，最右边算第0位。

### 输入

第一行是整数 t，表示数据组数。

每组输入数据是一行，三个整数 n,i和j。

### 输出

对每组输入数据，按十六进制输出结果。

### 样例输入

2

23 3 5

7 0 2

### 样例输出

30

3

## 编程题＃9：人群的排序和分类

**来源: 北京大学在线程序评测系统POJ (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)**

**总时间限制: 1000ms 内存限制: 1024kB**

### 描述

对人群按照输入的信息进行排序和分类。

#include <iostream>

#include <set>

#include <iterator>

#include <algorithm>

using namespace std;

// 在此处补充你的代码

int main()

{

int t;

cin >> t;

set<A\*,Comp> ct;

while( t -- ) {

int n;

cin >> n;

ct.clear();

for( int i = 0;i < n; ++i) {

char c; int k;

cin >> c >> k;

if( c == 'A')

ct.insert(new A(k));

else

ct.insert(new B(k));

}

for\_each(ct.begin(),ct.end(),Print);

cout << "\*\*\*\*" << endl;

}

}

### 输入

第一行是整数t,表明一共t组数据. t < 20

对每组数据：

第一行是整数n,表示下面一共有n行。 0 < n < 100

下面的每行代表一个人。每行以一个字母开头，代表该人所属的类别，然后跟着一个整数，代表年龄。字母只会是 'A‘或‘B' 。整数范围0到100。数据保证年龄都不相同。

### 输出

对每组输入数据，将这些人按年龄从小到大输出。每个人先输出类别，再输出年龄。每组数据的末尾加一行 "\*\*\*\*"

### 样例输入

2

4

A 3

B 4

A 5

B 6

3

A 4

A 3

A 2

### 样例输出

A 3

B 4

A 5

B 6

\*\*\*\*

A 2

A 3

A 4

\*\*\*\*

## 编程题＃10：输出指定结果二

**来源: 北京大学在线程序评测系统POJ (Coursera声明：在POJ上完成的习题将不会计入Coursera的最后成绩。)**

**总时间限制: 1000ms 内存限制: 1024kB**

### 描述

通过填空使得程序输出的结果符合下面的要求。

#include <iostream>

#include <map>

using namespace std;

// 在此处补充你的代码

int A::count = 0;

void func(B b) { }

int main()

{

A a1(5),a2;

cout << A::count << endl;

B b1(4);

cout << A::count << endl;

func(b1);

cout << A::count << endl;

A \* pa = new B(4);

cout << A::count << endl;

delete pa;

cout << A::count << endl;

return 0;

}

### 输入

不需要输入。

### 输出

使得程序的输出结果是：

2

3

B::destructor

A::destructor

3

4

B::destructor

A::destructor

3

B::destructor

A::destructor

A::destructor

A::destructor