9521 返回什么才好呢
9525 Big & Base 封闭类问题
14808 统计动物数量
16027　这个指针哪来的
3129 魔兽世界之一：备战

****14808.cpp:

```
#include <iostream>
using namespace std;
//your code starts here
class Animal {
    public:
        static int number;
        Animal();
        virtual ~Animal();
        Animal(const Animal & a);
};
int Animal::number = 0;
Animal::Animal() { ++ number; }
Animal::~Animal() { -- number; }
Animal::Animal(const Animal & a) { ++ number; }

class Dog:public Animal
{
    public:
        static int number;
        Dog();
        ~Dog();
        Dog(const Dog & a);
};
int Dog::number = 0;
Dog::Dog() { ++ number; }
Dog::~Dog() { -- number; }
Dog::Dog(const Dog & a) { ++ number; }

class Cat:public Animal
{
    public:
        static int number;
        Cat();
        ~Cat();
        Cat(const Cat & a);
};
```

```
int Cat::number = 0;
Cat::Cat() { ++ number; }
Cat::~Cat() { -- number; }
Cat::Cat(const Cat & a) { ++ number; }
//your code ends here
void print() {
    cout << Animal::number << " animals in the zoo, " << Dog::number << " of them
are dogs, " << Cat::number << " of them are cats" << endl;
}

int main() {
    print();
    Dog d1, d2;
    Cat c1;
    print();
    Dog* d3 = new Dog();
    Animal* c2 = new Cat;
    Cat* c3 = new Cat;
    print();
    delete c3;
    delete c2;
    delete d3;
    print();
}
```

****16027.cpp:

```
#include <iostream>
using namespace std;

struct A
{
    int v;
    A(int vv):v(vv) { }
//your code starts here
    const A * getPointer() const {
        return this;
    }
//your code ends here
};

int main()
{
```

```
    const A a(10);
    const A * p = a.getPointer();
    cout << p->v << endl;
    return 0;
}
```

****3129.cpp:

```
// by Guo Wei
#include <iostream>
#include <stdio.h>
using namespace std;
#define WARRIOR_NUM 5
/*
char * CWarrior::names[WARRIOR_NUM] = { "dragon","ninja","iceman","lion","wolf" };
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。
*/
class CHeadquarter;
class CWarrior
{
    private:
        CHeadquarter * pHeadquarter;
        int kindNo; //武士的种类编号 0 dragon 1 ninja 2 iceman 3 lion 4 wolf
        int nNo;
    public:
        static char * names[WARRIOR_NUM];
        static int InitialLifeValue [WARRIOR_NUM];
        CWarrior( CHeadquarter * p,int nNo_,int kindNo_ );
        void PrintResult(int nTime);
};
class CHeadquarter
{
    private:
        int totalLifeValue;
        bool bStopped;
        int totalWarriorNum;
        int color;
        int curMakingSeqIdx; //当前要制造的武士是制造序列中的第几个
        int warriorNum[WARRIOR_NUM]; //存放每种武士的数量
        CWarrior * pWarriors[1000];
    public:
```

```
        friend class CWarrior;
        static int makingSeq[2][WARRIOR_NUM]; //武士的制作顺序序列
        void Init(int color_, int lv);
        ~CHeadquarter () ;
        int Produce(int nTime);
        void GetColor( char * szColor);

};
CWarrior::CWarrior(         CHeadquarter         *         p,int         nNo_,int
kindNo_ ):nNo(nNo_),kindNo(kindNo_),pHeadquarter(p) {  }
void CWarrior::PrintResult(int nTime)
{
        char szColor[20];
        pHeadquarter->GetColor(szColor);
        printf("%03d %s %s %d born with strength %d,%d %s in %s headquarter\n"
    ,
                nTime, szColor, names[kindNo], nNo, InitialLifeValue[kindNo],
                pHeadquarter->warriorNum[kindNo],names[kindNo],szColor);
}
void CHeadquarter::Init(int color_, int lv)
{
    color = color_;
    totalLifeValue = lv;
    totalWarriorNum = 0;
    bStopped = false;
    curMakingSeqIdx = 0;
    for( int i = 0;i < WARRIOR_NUM;i ++ )
        warriorNum[i] = 0;
}
CHeadquarter::~CHeadquarter () {
    for( int i = 0;i < totalWarriorNum;i ++ )
        delete pWarriors[i];
}
int CHeadquarter::Produce(int nTime)
{

    if( bStopped )
        return 0;
    int nSearchingTimes = 0;
    while(    CWarrior::InitialLifeValue[makingSeq[color][curMakingSeqIdx]]    >
totalLifeValue &&
        nSearchingTimes < WARRIOR_NUM ) {
        curMakingSeqIdx = ( curMakingSeqIdx + 1 ) % WARRIOR_NUM ;
        nSearchingTimes ++;
```

```
    }
    int kindNo = makingSeq[color][curMakingSeqIdx];
    if( CWarrior::InitialLifeValue[kindNo] > totalLifeValue ) {
        bStopped = true;
        if( color == 0)
            printf("%03d red headquarter stops making warriors\n",nTime);
        else
            printf("%03d blue headquarter stops making warriors\n",nTime);
        return 0;
    }
    totalLifeValue -= CWarrior::InitialLifeValue[kindNo];
    curMakingSeqIdx = ( curMakingSeqIdx + 1 ) % WARRIOR_NUM ;
    pWarriors[totalWarriorNum] = new CWarrior( this,totalWarriorNum+1,kindNo);
    warriorNum[kindNo]++;
    pWarriors[totalWarriorNum]->PrintResult(nTime);
    totalWarriorNum ++;
    return 1;
}
void CHeadquarter::GetColor( char * szColor)
{
    if( color == 0)
        strcpy(szColor,"red");
    else
        strcpy(szColor,"blue");
}

char * CWarrior::names[WARRIOR_NUM] = { "dragon","ninja","iceman","lion","wolf" };
int CWarrior::InitialLifeValue [WARRIOR_NUM];
int CHeadquarter::makingSeq[2][WARRIOR_NUM] = { { 2,3,4,1,0 },{3,0,1,2,4} }; //
两个司令部武士的制作顺序序列

int main()
{
    int t;
    int m;
    CHeadquarter RedHead,BlueHead;
    scanf("%d",&t);
    int nCaseNo = 1;
    while ( t -- ) {
        printf("Case:%d\n",nCaseNo++);
        scanf("%d",&m);
        for( int i = 0;i < WARRIOR_NUM;i ++ )
            scanf("%d", & CWarrior::InitialLifeValue[i]);
        RedHead.Init(0,m);
```

```
        BlueHead.Init(1,m);
        int nTime = 0;
        while( true) {
            int tmp1 = RedHead.Produce(nTime);
            int tmp2 = BlueHead.Produce(nTime);
            if( tmp1 == 0 && tmp2 == 0)
                break;
            nTime ++;
        }
    }
    return 0;
}
```

****9521.cpp:

```
/*
程序填空，使其按要求输出

输入：
多组数据，每组一行，是整数 m 和 n

输出：
先输出一行：
123
然后，对每组数据，输出两行，第一行是 m,第二行是 n

输入样例
2 3
4 5
输出样例
123
2
3
4
5

*/

#include <iostream>
using namespace std;
class A {
public:
```

```
    int val;

    A(int
//your code start here
     n = 123  ) { val = n;};
    A &  GetObj() {
        return * this;
    }
//  your code ends here
};
int main()
{
    int m,n;
    A a;
    cout << a.val << endl;
    while(cin >> m >> n) {
        a.GetObj() = m;
        cout << a.val << endl;
        a.GetObj() = A(n);
        cout << a.val<< endl;
    }
    return 0;
}
```

****9525.cpp:

/*
程序填空，输出指定结果

输入：

多组数据，每组一行，是一个整数

输出
对每组数据，输出两行，每行把输入的整数打印两遍

样例输入
3
4
样例输出
3,3

```
3,3
4,4
4,4


*/
#include <iostream>
#include <string>
using namespace std;
class Base {
public:
    int k;
    Base(int n):k(n) { }
};
class Big
{
public:
    int v;
    Base b;
//your code starts here
    Big (int n):v(n),b(n){ }
    Big (const Big & bb):v(bb.v),b(bb.v){ }
//your code ends here

};
int main()
{
    int n;
    while(cin >>n) {
        Big a1(n);
        Big a2 = a1;
        cout << a1.v << "," << a1.b.k << endl;
        cout << a2.v << "," << a2.b.k << endl;
    }
}
```