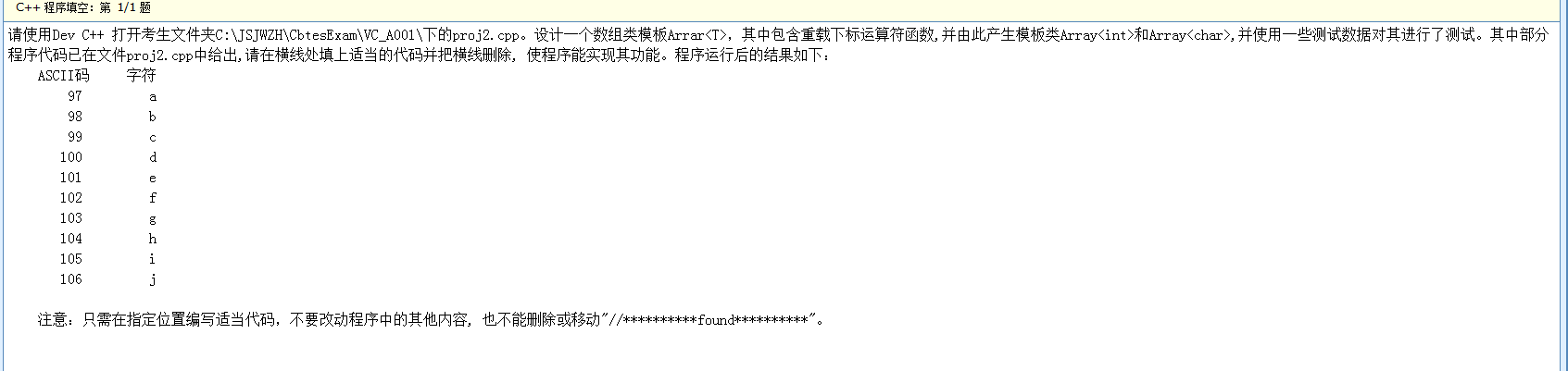
**填空题**



#include <iostream>

#include <iomanip>

using namespace std;

template <class T>

class Array

{

T \*elems ;

int size ;

public:

Array(int s) ;

~Array() ;

T& operator[](int) ; // 重载下标运算符

void operator=(T) ; // 重载等号运算符

};

template <class T> Array<T>::Array(int s)

{

size = s ;

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

elems = \_\_new\_\_ T[size] ;

for(int i = 0 ; i < size ; i++) elems[i] = 0 ;

}

template <class T> Array<T>::~Array()

{

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

\_\_\_delete elems\_\_\_\_\_\_\_\_\_ ;

}

template <class T> T& Array<T>::operator[](int index)

{

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_return elems[index]\_\_\_\_\_\_\_\_\_\_\_\_ ;

}

template <class T>void Array<T>::operator=(T temp)

{

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

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

elems[i] = \_temp\_\_\_ ;

}

int main()

{

int i, n = 10 ;

Array<int> arr1(n) ; // 产生整数型模板类及其对象arr1

Array<char> arr2(n) ; // 产生字符型模板类及其对象arr2

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

{

arr1[i] = 'a' + i ; // 调用重载运算符

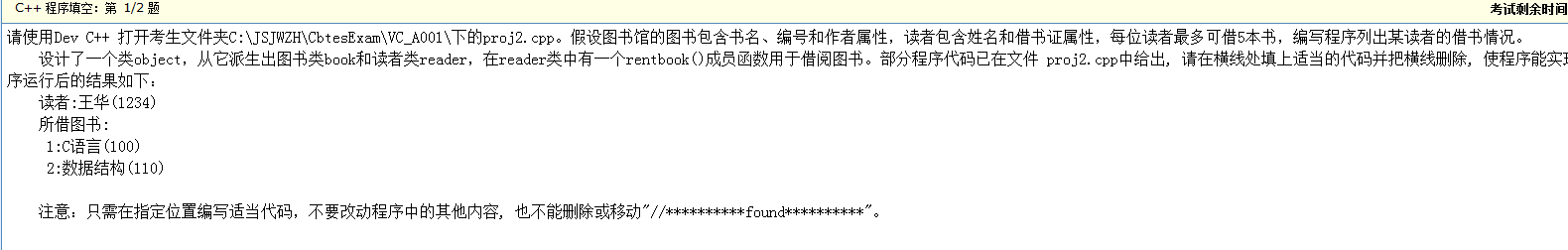
arr2[i] = 'a' + i ;

}

cout << " ASCII码 字符" << endl ;

for( i = 0 ; i < n ; i++) cout << setw(8) << arr1[i] << setw(8) << arr2[i] << endl ;

}



#include <iostream>

using namespace std;

#include <string.h>

class object

{

char name[20] ;

int no ;

public:

object() {}

object(const char na[], int n)

{

strcpy(name, na) ; no = n ;

}

void show()

{

cout << name << "(" << no << ")" ;

}

};

class book:public object

{

char author[10] ;

public:

book() {}

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

book(const char na[], int n, const char auth[]):object(na,n)

{

strcpy(author, auth) ;

}

void showbook()

{

show() ;

cout << " 作者:" << author ;

}

};

class reader:public object

{

book rent[5] ;

int top ;

public:

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

reader(const char na[], int n):object(na,n) {top = 0; }

void rentbook(book &b)

{

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

rent[top]=b ;

top++ ;

}

void showreader()

{

cout << "读者:" ; show() ;

cout << endl << "所借图书:" << endl ;

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

{

cout << " " << i+1 << ":" ;

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

rent[i].show();

cout << endl ;

}

}

};

int main()

{

book b1("C语言",100,"谭浩强"), b2("数据结构",110,"严蔚敏") ;

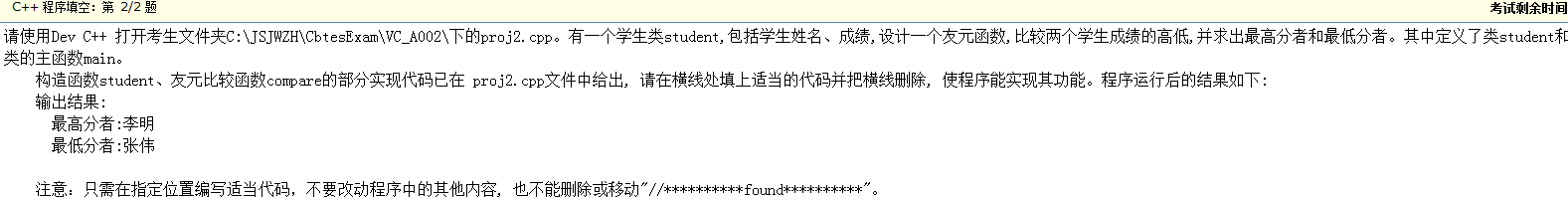
reader r1("王华",1234) ;

r1.rentbook(b1) ;

r1.rentbook(b2) ;

r1.showreader() ;

}



#include <iostream>

#include <string.h>

using namespace std;

class student

{

char name[10] ;

int deg ;

public:

student(const char na[], int d)

{

strcpy(name, na) ;

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

deg=d;;

}

char \*getname() { return name ; }

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

friend int compare(student &s1,student &s2)

{

if(s1.deg > s2.deg) return 1 ;

else if(s1.deg == s2.deg) return 0 ;

else return -1 ;

}

};

int main()

{

student st[] = {student("王华",78), student("李明",92), student("张伟",62), student("孙强",88)} ;

int min = 0 ; // 存放最低分的位置

int max = 0 ; // 存放最高分的位置

for(int i = 1 ; i < 4 ; i++)

{

if(compare(st[max], st[i]) == -1)

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

max=i;

else if(compare(st[min], st[i]) == 1)

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

min=i;

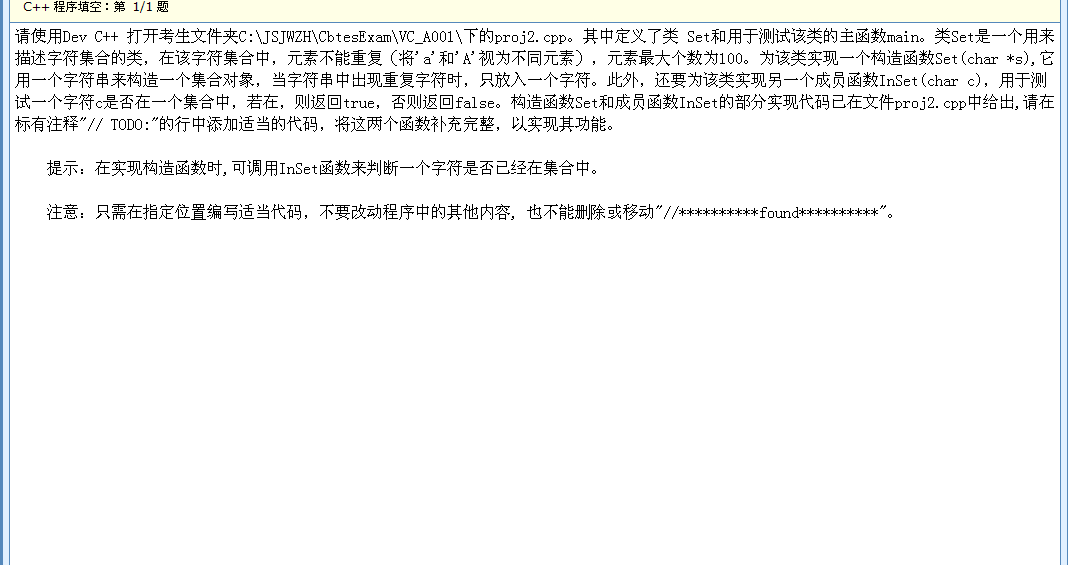
}

cout << "输出结果:" << endl ;

cout << " 最高分者:" << st[max].getname() << endl ;

cout << " 最低分者:" << st[min].getname() << endl ;

}



// proj2.cpp

#include <iostream>

using namespace std;

const int MAXNUM = 100;

class Set {

private:

int num; // 元素个数

char setdata[MAXNUM]; // 字符数组，用于存储集合元素

public:

Set(char \*s); // 构造函数，用字符串s构造一个集合对象

bool InSet(char c); // 判断一个字符c是否在集合中，若在，返回true，否则返回false

void Print() const; // 输出集合中所有元素

};

Set::Set(char \*s)

{

num = 0;

while (\*s){

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

if (\*s!=setdata[num]) // TODO: 添加代码，测试元素在集合中不存在

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

setdata[num++]=\*s; // TODO: 添加一条语句，加入元素至集合中

s++;

}

}

bool Set::InSet(char c)

{

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

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

if (setdata[i]==c) // TODO: 添加代码，测试元素c是否与集合中某元素相同

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

return true; // TODO: 添加一条语句，进行相应处理

return false;

}

void Set::Print() const

{

cout << "Set elements: " << endl;

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

cout << setdata[i] << ' ';

cout << endl;

}

int main()

{

char s[MAXNUM];

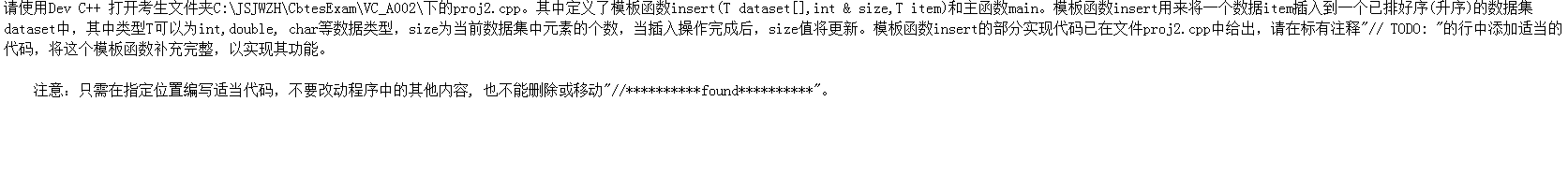
cin.getline(s, MAXNUM-1); // 从标准输入中读入一行

Set setobj(s); // 构造对象setobj

setobj.Print(); // 显示对象setobj中内容

return 0;

}



// proj2.cpp

#include <iostream>

using namespace std;

//请在该部分插入insert函数模板的实现

template <typename T>

void insert(T setdata[], int &size, T item)

{

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

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

if (item<setdata[i]) { // TODO: 添加代码，判断查找元素的插入位置

for (int j = i; j < size; j++)

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

setdata[size-j+i]=setdata[size-j+i-1]; // TODO: 添加一条语句，将插入位置后的所有元素往后移动一个位置

// 提示：移动元素应从最后一个元素开始移动

setdata[i] = item; // 插入该元素

size++;

return;

}

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

setdata[size] = item; //有错误 TODO: 添加一条语句，将元素加到最后一个位置上

size++;

return;

}

int main()

{

int idata[10] = { 22, 35, 56, 128 }, iitem, isize = 4, dsize = 4, i;

double ddata[10] = { 25.1, 33.5, 48.9, 75.3 }, ditem;

cout << "Please input one integer number for inserting:";

cin >> iitem;

insert(idata, isize, iitem);

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

cout << idata[i] << ' ';

cout << endl;

cout << "Please input one double number for inserting:";

cin >> ditem;

insert(ddata, dsize, ditem);

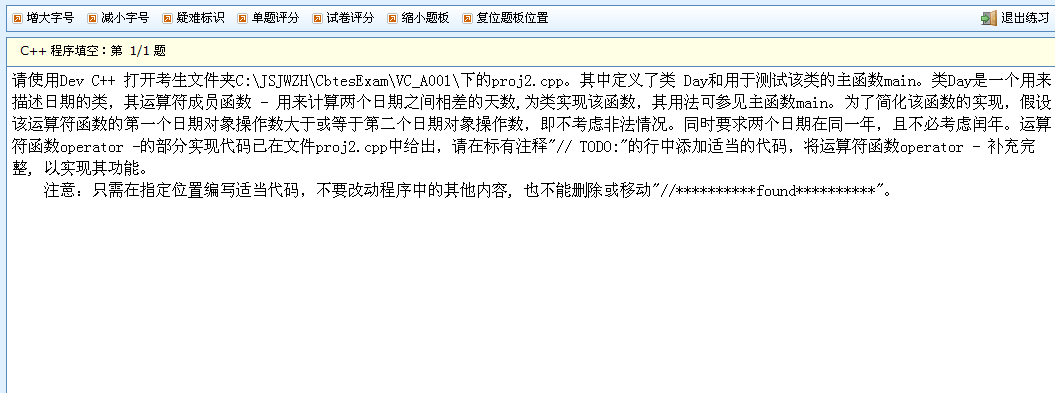
for (i = 0; i < dsize; i++)

cout << ddata[i] << ' ';

cout << endl;

return 0;

}



// proj2.cpp

#include <iostream>

using namespace std;

class Day {

private:

int year;

int month;

int day;

public:

Day(int y=2000, int m=1, int d=1); // 构造函数

int operator - (const Day &d); // 重载运算符 -

void Print() const; // 输出日期

};

Day::Day(int y, int m, int d)

{

year = y;

month = m;

day = d;

}

int Day::operator - (const Day &d) // 实现运算符函数 -

{

int diffs, m;

int monthday[13] = { // 存放每月的天数

0,31,28,31,30,31,30,31,31,30,31,30,31

};

if (year == d.year && month == d.month) { // 若两日期在同年同月内, 则计算其相差的天数

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

diffs =day-d.day; // TODO: 在此增加代码

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

return diffs ; // TODO: 在此增加一条语句

}

diffs = monthday[d.month] - d.day + day; // 计算除整月以外相差的天数

for (m = d.month+1; m < month; m++)

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

diffs+=monthday[m] ; // TODO: 在此增加一条语句, 依次计算两个日期间相差的整月天数

return diffs;

}

void Day::Print( ) const

{

cout << "Year = " << year << ", Month = " << month << " , Day = " << day << endl;

}

int main()

{

int y,m,d;

cin >> y >> m >> d;

Day d1(y, m, d);

cin >> y >> m >> d;

Day d2(y, m, d);

int diff\_days;

d1.Print();

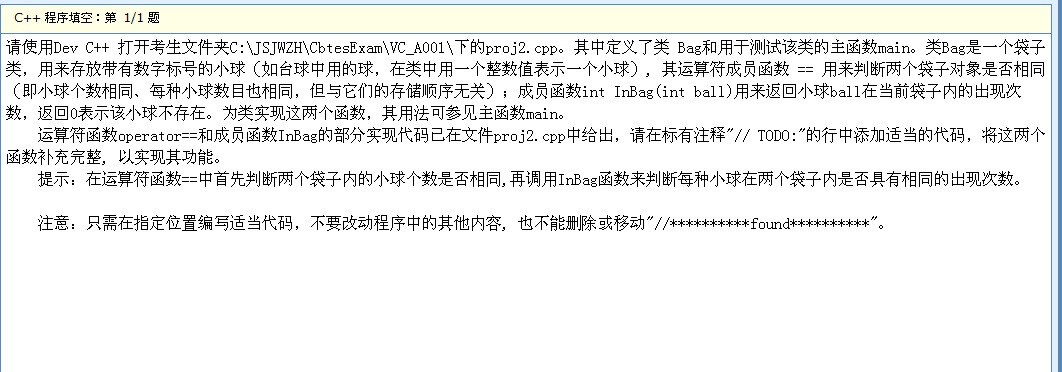
d2.Print();

diff\_days = d1 - d2; // 计算d1和d2相差天数

cout << "Difference days: " << diff\_days << endl;

return 0;

}



// proj2.cpp

#include <iostream>

using namespace std;

const int MAXNUM = 100;

class Bag {

private:

int num;

int bag[MAXNUM];

public:

Bag(int m[], int n=0); // 构造函数

bool operator == (Bag &b); // 重载运算符==

int InBag(int ball); // 某一小球在袋子内的出现次数,返回0表示不存在

};

Bag::Bag(int m[], int n)

{

if(n > MAXNUM) {

cerr << "Too many members\n";

exit(-1);

}

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

bag[i] = m[i];

num = n;

}

bool Bag::operator == (Bag &b) // 实现运算符函数==

{

if (num != b.num) // 元素个数不同

return false;

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

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

if (InBag(bag[i])!=b.InBag(bag[i])) // TODO: 加入条件, 判断当前袋子中每个元素在当前袋子和袋子b中是否出现次数不同

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

return false; // TODO: 加入一条语句

return true;

}

int Bag::InBag(int ball)

{

int count = 0;

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

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

if (bag[i]=ball) // TODO: 加入条件, 判断小球ball是否与当前袋子中某一元素相同

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

count++ ; // TODO: 加入一条语句

return count;

}

int main()

{

int data[MAXNUM], n, i;

cin >> n;

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

cin >> data[i];

Bag b1(data, n); // 创建袋子对象b1

cin >> n;

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

cin >> data[i];

Bag b2(data, n); // 创建袋子对象b2

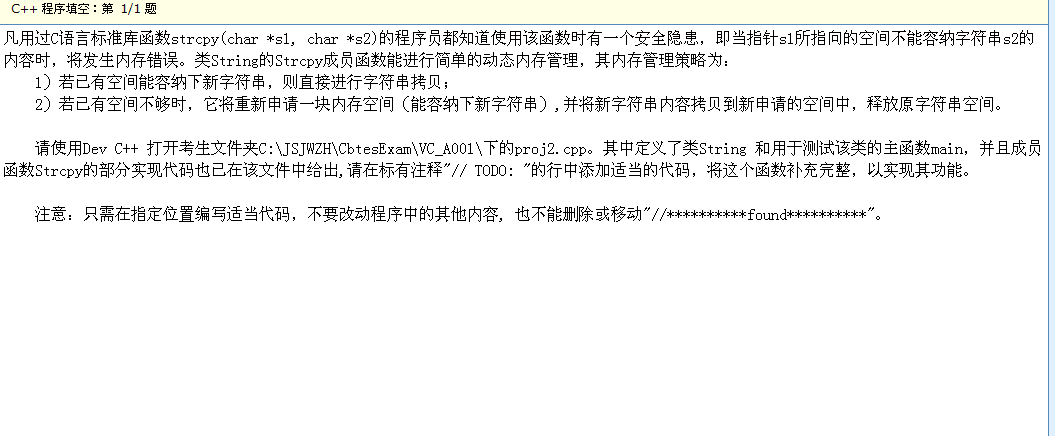
if( b1 == b2) // 测试b1和b2是否相同

cout << "Bag b1 is same with Bag b2\n";

else

cout << "Bag b1 is not same with Bag b2\n";

return 0;}

// proj2.cpp

#include <string.h>

#include <iostream>

using namespace std;

class String {

private:

int size; // 缓冲区大小

char \*buf; // 缓冲区

public:

String(int bufsize);

void Strcpy(char \*s); // 将字符串s复制到buf中

void Print() const;

~String() { if (buf != NULL) delete buf; }

};

String::String(int bufsize)

{

size = bufsize;

buf = new char[size];

\*buf = '\0';

}

void String::Strcpy(char \*s)

{

char \*p,\*q;

int len = strlen(s);

if (len+1 > size) {

size = len+1;

p = q = new char[size];

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

while(\*(q++)=\*(s++)); // TODO: 添加代码将字符串s拷贝到字符指针q中

delete [] buf;

buf = p;

}

else {

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

for(p=buf;\*p=\*s;p++,s++); // TODO: 添加代码将字符串s拷贝到buf中

}

}

void String::Print() const

{

cout << size << '\t' << buf << endl;

}

int main()

{

char s[100];

String str(32);

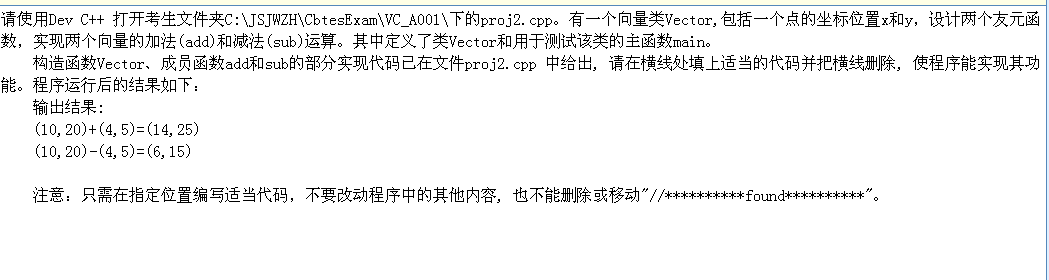
cin.getline(s, 99);

str.Strcpy(s);

str.Print();

return 0;

}



#include <iostream>

using namespace std;

class Vector

{

int x,y ;

public:

Vector() {}

Vector(int i, int j) { x = i ; y = j ; }

void display()

{

cout << "(" << x << "," << y << ")" ;

}

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

friend Vector add(Vector &v1,Vector &v2)

{

Vector v ;

v.x = v1.x + v2.x ;

v.y = v1.y + v2.y ;

return v ;

}

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

friend Vector sub(Vector &v1,Vector &v2)

{

Vector v ;

v.x = v1.x - v2.x ;

v.y = v1.y - v2.y ;

return v ;

}

};

int main()

{

Vector v1(10,20), v2(4,5), v3 ;

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

v3 = add(v1,v2) ;

cout << "输出结果:" << endl ;

cout << " " ; v1.display() ; cout << "+" ; v2.display() ;

cout << "=" ; v3.display() ; cout << endl ;

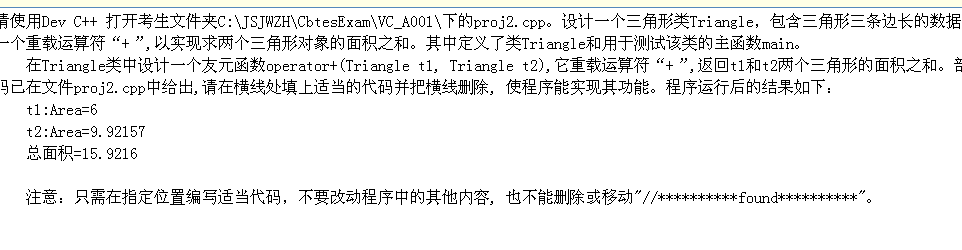
//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

v3 = sub(v1,v2) ;

cout << " " ; v1.display() ; cout << "-" ; v2.display() ;

cout << "=" ; v3.display() ; cout << endl ;

}



#include <iostream>

#include <math.h>

using namespace std;

class Triangle

{

int x,y,z ;

double area ;

public:

Triangle(int i, int j, int k)

{

double s ;

x = i ; y = j ; z = k ;

s = (x + y + z) / 2.0 ;

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

area = sqrt(s \* (s - x) \* (s - y) \* (s - z)) ;

}

void display()

{

cout << "Area=" << area << endl ;

}

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

friend double operator+(Triangle x,Triangle y)

{

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

return x.area+y.area;

}

};

int main()

{

Triangle t1(3,4,5), t2(4,5,6) ;

double s ;

cout << "t1:";t1.display() ;

cout << "t2:";t2.display() ;

//\*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

s = t1+t2 ;

cout << "总面积=" << s << endl ;

}