**实验1:复习**

**姓名\_\_李飞飞\_\_ 班级\_\_软工2206\_ 学号\_202105710309\_**

* **请阅读此说明：实验1最高120分；其中20分为附加，可选做；做完实验后请按要求将代码和截图贴入该文档。然后将此文档、源代码文件（.cpp）打包上传到学习通。**

**备注：1,2题 选做一题即可。**

1. **数组+函数 基础复习部分：（20分）**

**有如下主函数框架：**

**int main()**

**{ unsigned n;**

**int \*p;**

**while(cin>>n)**

**{**

**p= InitArray(n);**

**//初始化一个规模为n的一维整型数组，并给出随机互异的值进行初始化。**

**Display(p,n); //显示数据的初始化情况**

**SortArray(p,n);//根据每个数包含的百位和个位组成的新数值进行升序整理。**

**Display(p,n); //显示排序后的结果**

**DestroyArray(p); //根据需要使用**

**}**

**return 0;**

**}**

* **实验要求：**

**❶根据主函数工作要求，设计对应的函数原型并将他们实现；**

**❷将所有实现的函数与主函数进行装配测试，并补充完全所有程序运行需要的内容。**

**❸给出测试的截图。分别使用n为12； n为123；n为1234的数据规模**

* **实验提交：**

**将完整源代码+适当注释和测试截图 黏贴在下面。**

* + **代码黏贴处：**
  + **截图黏贴处：**

1. **数组+函数 提高复习部分：（20分）**

**利用1的处理模型，考虑以下测试需求，即程序可能接收多组测试，每组测试由组数，每组的数据个数，每组数组值 构成；最后输出对当前组升序排列的结果。**

**输入测试样本：**

**3**

**3 122 131 133**

**5 4210 2010 3241 1623 232**

**4 235 14 368 201**

**测试输出样本：**

**131 122 133**

**2010 4210 3241 232 1623**

**14 201 235 368**

* **实验要求：**

**❶ 以题1的实现为基础，适当调整，完成题2的工作要求。**

* **实验提交：**

**将完整的源代码和测试截图 粘贴在下面。 测试可使用样例测试。**

* + **代码黏贴处：**

#include<iostream>

#include<algorithm>

#include<unordered\_set>

#include<ctime>

#include<cstdlib>

using namespace std;

#define ll long long

#define endl '\n'

#define fast ios::sync\_with\_stdio(false);cin.tie(0);cout.tie(0);

ll n,m;

int \*p;

int get(int &k){

    int ans=0;

    ans+=k%10;

    ans+=k/100%10\*10;

    return ans;

}

//初始化数组为n个不同元素

void InitArray(int n){

    p=new int[n];

    unordered\_set<int> st;

    while (st.size()!=n){

        int k=rand();

        st.insert(k);

    }

    int now=0;

    for(auto t:st)

        p[now++]=t;

}

//打印数组

void Display(int \*p,int n){

    for(int i=0;i<n;i++){

        cout<<p[i]<<" ";

        if(n%20&&n)cout<<endl;

    }

}

//排序数组

void SortArray(int \*p,int n){

    sort(p,p+n,[](int a,int b){

        int sa=get(a),sb=get(b);

        return sa>sb;

    });

}

//销毁数组

void DestroyArray(int \*p){

    delete []p;

    p=nullptr;

}

void solve(){

    cin>>n;

    p=new int[n];

    for(int i=0;i<n;i++)cin>>p[i];

    SortArray(p,n);

    Display(p,n);

    DestroyArray(p);

}

int main(){

    srand(time(0));

    fast

    int T=1;

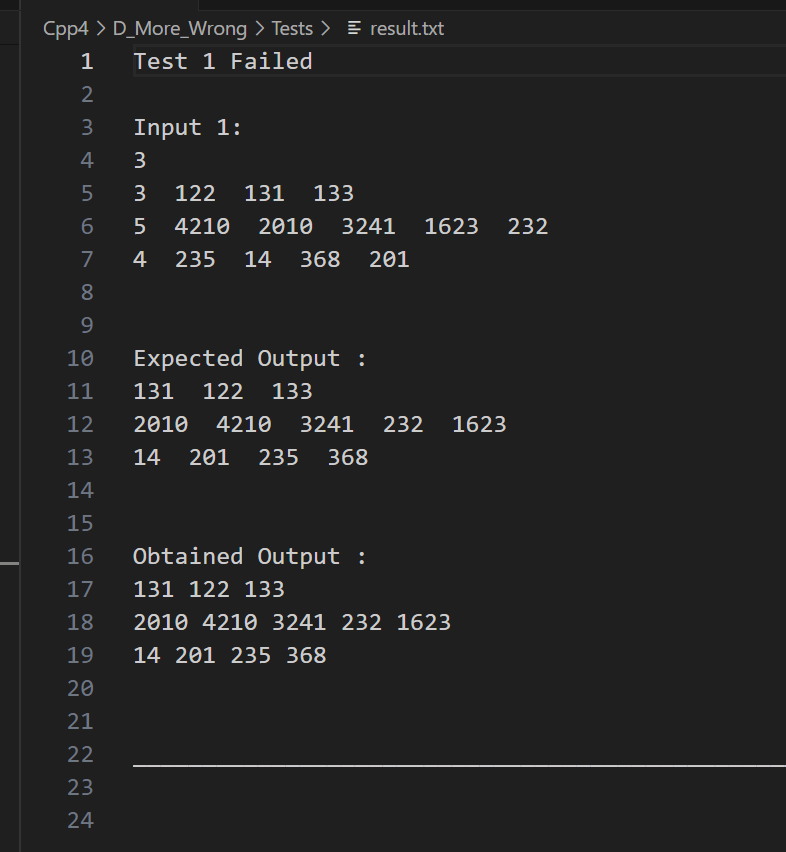
    cin>>T;

    while(T--) solve();

    return 0;

}

* + **截图黏贴处：**

****

**3、数组+函数 提高复习部分：（20分）**

**有如下主函数框架：**

**int main(){**

**const int n=2;**

**const int k=3**

**const int m=4;**

**int a[n][k], b[k][m];**

**//请补充c的定义**

**InitMatrix(a,n,k); //初始化矩阵a**

**InitMatrix(b,k,m); //初始化矩阵b**

**c=Multiply(a,b,n,k,k,m); //计算aXb ，可以省略一个参数k**

**Display(c); //输出计算结果, 有可能还需要其他参数，根据自己定义的c调整。**

**return 0;**

**}**

* **实验要求：**

**❶根据主函数的框架理解程序的工作目的，请你根据调用需求设计对应函数的原型并将他们进行实现；**

**❷将你设计的所有函数与主函数进行装配测试，并补充完全所有程序运行需要的内容。**

**❸给出测试的截图。可以使用以下测试数据，也可以自行设计。**

**a (1,2,3; 4,5,6)**

**b(1,2,3,4;2,3,4,5;3,4,5,6)**

**❹思考：若a,b的规模可以由用户设定，即程序可以接受定制的cin>>n>>k>>m; 那么你的设计需要做哪些调整？给出调整后的源代码和测试截图。（附加20分，）**

* **实验提交：**
  + **代码黏贴处：**

#include<iostream>

#include<ctime>

#include<cstdlib>

using namespace std;

#define ll long long

#define endl '\n'

ll n, m, k;

//随机生成

void InitMatrix(int\*\* &p, int n, int m) {

    p = new int\* [n];

    for (int i = 0; i < n; i++)

        p[i] = new int[m];

    // for (int i = 0; i < n; i++)

    //     for (int j = 0; j < m; j++)

    //         //防止乘法爆int

    //         p[i][j] = rand() % 10000;

}

//展示

inline void Display(int\*\* p, int n, int m) {

    cout << "展示一个矩阵：" << endl;

    for (int i = 0; i < n; i++) {

        for (int j = 0; j < m; j++)

            cout << p[i][j] << " ";

        cout << endl;

    }

    cout << endl;

}

//摧毁

void DestroyArray(int\*\* p, int n) {

    for (int i = 0; i < n; i++)

        delete[]p[i];

    delete[]p;

}

//矩阵乘法，生成n\*k矩阵

int\*\* Multiply(int\*\* pa, int\*\* pb, int n, int m, int k) {

    int\*\* p3=nullptr;

    InitMatrix(p3, n, k);

    for (int i = 0; i < n; i++)

        for (int j = 0; j < k; j++) {

            p3[i][j] = 0;

            for (int t = 0; t < m; t++)

                p3[i][j] += pa[i][t] \* pb[t][j];

        }

    return p3;

}

void solve() {

    // cin >> n >> m >> k;

    n=2,m=3,k=4;

    int\*\* p1=nullptr, \*\* p2=nullptr;

    p1 = new int\*[n];

    p2 = new int\*[m];

    // 为每一行分配内存并赋值

    p1[0] = new int[m]{1, 2, 3};

    p1[1] = new int[m]{4, 5, 6};

    p2[0] = new int[k]{1, 2, 3, 4};

    p2[1] = new int[k]{2, 3, 4, 5};

    p2[2] = new int[k]{3, 4, 5, 6};

    Display(p1, n, m);

    Display(p2, m, k);

    int\*\* p3 = Multiply(p1, p2, n, m, k);

    Display(p3, n, k);

    DestroyArray(p1, n); DestroyArray(p2, m);

    DestroyArray(p3, n);

}

int main() {

    srand(time(0));

    int T = 1;

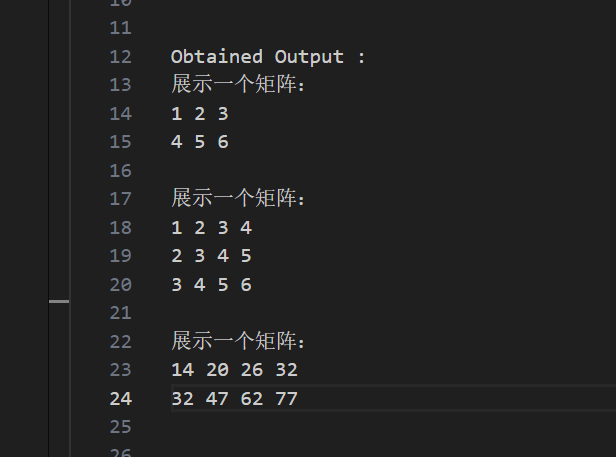
    //    cin>>T;

    while (T--) solve();

    return 0;

}

* + **截图黏贴处：**

****

**思考❹的**

* **代码黏贴处：**

#include<iostream>

#include<ctime>

#include<cstdlib>

using namespace std;

#define ll long long

#define endl '\n'

ll n, m, k;

//随机生成

void InitMatrix(int\*\* &p, int n, int m) {

    p = new int\* [n];

    for (int i = 0; i < n; i++)

        p[i] = new int[m];

    for (int i = 0; i < n; i++)

        for (int j = 0; j < m; j++)

            //防止乘法爆int

            p[i][j] = rand() % 10000;

}

//展示

inline void Display(int\*\* p, int n, int m) {

    cout << "展示一个矩阵：" << endl;

    for (int i = 0; i < n; i++) {

        for (int j = 0; j < m; j++)

            cout << p[i][j] << " ";

        cout << endl;

    }

    cout << endl;

}

//摧毁

void DestroyArray(int\*\* p, int n) {

    for (int i = 0; i < n; i++)

        delete[]p[i];

    delete[]p;

}

//矩阵乘法，生成n\*k矩阵

int\*\* Multiply(int\*\* pa, int\*\* pb, int n, int m, int k) {

    int\*\* p3=nullptr;

    InitMatrix(p3, n, k);

    for (int i = 0; i < n; i++)

        for (int j = 0; j < k; j++) {

            p3[i][j] = 0;

            for (int t = 0; t < m; t++)

                p3[i][j] += pa[i][t] \* pb[t][j];

        }

    return p3;

}

void solve() {

    cin >> n >> m >> k;

    int\*\* p1=nullptr, \*\* p2=nullptr;

    InitMatrix(p1, n, m); InitMatrix(p2, m, k);

    Display(p1, n, m); Display(p2, m, k);

    int\*\* p3 = Multiply(p1, p2, n, m, k);

    Display(p3, n, k);

    DestroyArray(p1, n); DestroyArray(p2, m);

    DestroyArray(p3, n);

}

int main() {

    srand(time(0));

    int T = 1;

    //    cin>>T;

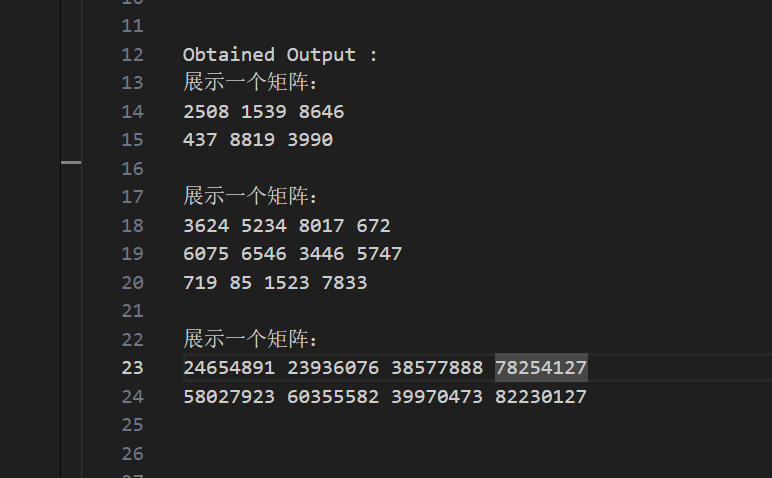
    while (T--) solve();

    return 0;

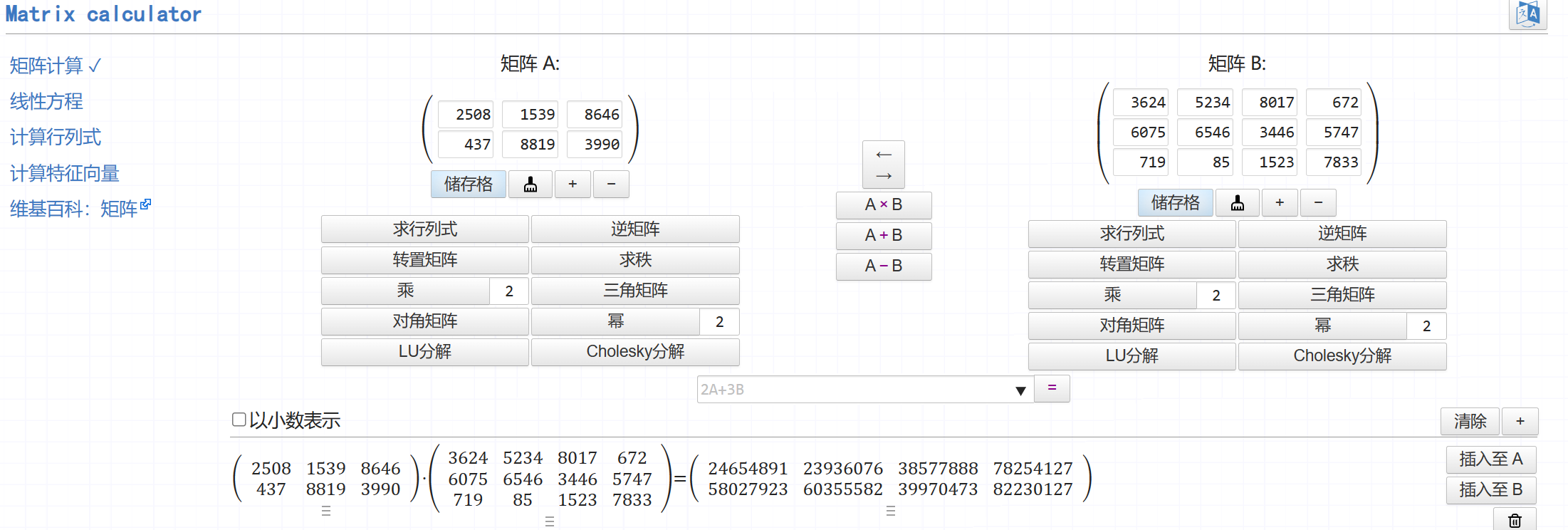
}

* **截图黏贴处：**

**矩阵大小是输入的2 3 4，矩阵数值是随机且不会爆int的数**

****

**试过了是对的**

****

**4、引用做形参和返回类型：（40分）**

**❶重做课堂例程：两个元素的交换 （10分）**

**#include<iostream>**

**using namespace std;**

**//swap的声明**

**int main()**

**{ int a,b;**

**cin>>a>>b;**

**cout<<a<<b<<endl;**

**/\*int t;**

**t=a;**

**a=b;**

**b=t;\*/**

**//交换函数swap的调用**

**cout<<a<<b<<endl;**

**return 0;**

**}**

**//swap的定义**

**考虑三种swap的实现：**

**swap(int,int);**

**swap(int\*,int\*);**

**swap(int&,int&);**

**体会引用形参和其他类型参数的差异。**

void swap1(int a,int b){

    //无法交换

}

void swap2(int &a,int &b){

    int t=a;

    a=b;b=t;

}

void swap3(int \*a,int \*b){

    int t=\*a;

    \*a=\*b;

    \*b=t;

}

**❷完成课堂例题：学生数据的输入输出处理（30分）**

**学生结构如下所示：**

**struct student{**

**int id;**

**char name[20];**

**int scores[3];**

**};**

**现需要处理一个10人的班级的学生数据，要求程序完成：a、学生信息的输入，b、学生成绩三门课最高分获得者的求解和总分最高获得者的求解 fun()；c、完成学生信息的输出。**

**//两种输入原型**

**void inputStu(student \*); //a1**

**void inputStu(student&); //a2**

**//三种输出原型**

**void outputStu(student); //c1考虑为什么输入不能使用这种类型的形参？**

**void outputStu(student\*); //c2**

**void outputStu(student&);//c3**

**//fun的原型**

**int main()**

**{ const int n=10;**

**student aclass[n];**

**int i;**

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

**// a、inputStu的调用; 分别使用两个原型测试**

**//b、fun的调用**

**//输出求解结果，单科最高分的学号、姓名、分值，总分最高的学号、姓名、总分值。**

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

**//c、outputStu的调用; 分别使用三个原型测试**

**return 0;**

**}**

* + **代码黏贴处：**

**//给出对应的inputStu的实现**

**a1:**

void inputStu(student \*stu){

    cin>>stu->id>>stu->name;

    for(int i=0;i<3;i++)

    cin>>stu->scores[i];

}

**a2:**

void inputStu(student &stu){

    cin>>stu.id;

    cin>>stu.name;

    for(int i=0;i<3;i++)

    cin>>stu.scores[i];

}

**//给出对应outputStu的实现**

**c1:**

void outputStu(const student stu){

    cout<<"id:"<<stu.id<<"  学生姓名："<<stu.name<<endl;

    cout<<"三门课成绩：";

    cout<<stu.scores[0]<<" "<<stu.scores[1]<<" "<<stu.scores[2]<<endl;

}

**c2:**

void outputStu(const student \*stu){

    cout<<"id:"<<stu->id<<"  学生姓名："<<stu->name<<endl;

    cout<<"三门课成绩：";

    cout<<stu->scores[0]<<" "<<stu->scores[1]<<" "<<stu->scores[2]<<endl;

}

**c3:**

void outputStu(const student &stu){

    cout<<"id:"<<stu.id<<"  学生姓名："<<stu.name<<endl;

    cout<<"三门课成绩：";

    cout<<stu.scores[0]<<" "<<stu.scores[1]<<" "<<stu.scores[2]<<endl;

}

**//fun函数的原型设计和调用，考虑使用引用形参返回三门功课的单科最高分获得者，以及总分最高获得者。**

vector<student\*> fun(vector<student> &q){

    //分别表示高分获得者 三门课最高分获得者

    student \*ma=nullptr;

    student \*ma1=nullptr,\*ma2=nullptr,\*ma3=nullptr;

    //引用形参操作每个对象

    for(auto &t:q){

        int sum=accumulate(t.scores,t.scores+3,0);

        if(ma==nullptr||sum>accumulate(ma->scores,ma->scores+3,0))

            ma=&t;

        if(ma1==nullptr||ma1->scores[0]<t.scores[0])

            ma1=&t;

        if(ma2==nullptr||ma2->scores[1]<t.scores[1])

            ma2=&t;

        if(ma3==nullptr||ma3->scores[2]<t.scores[2])

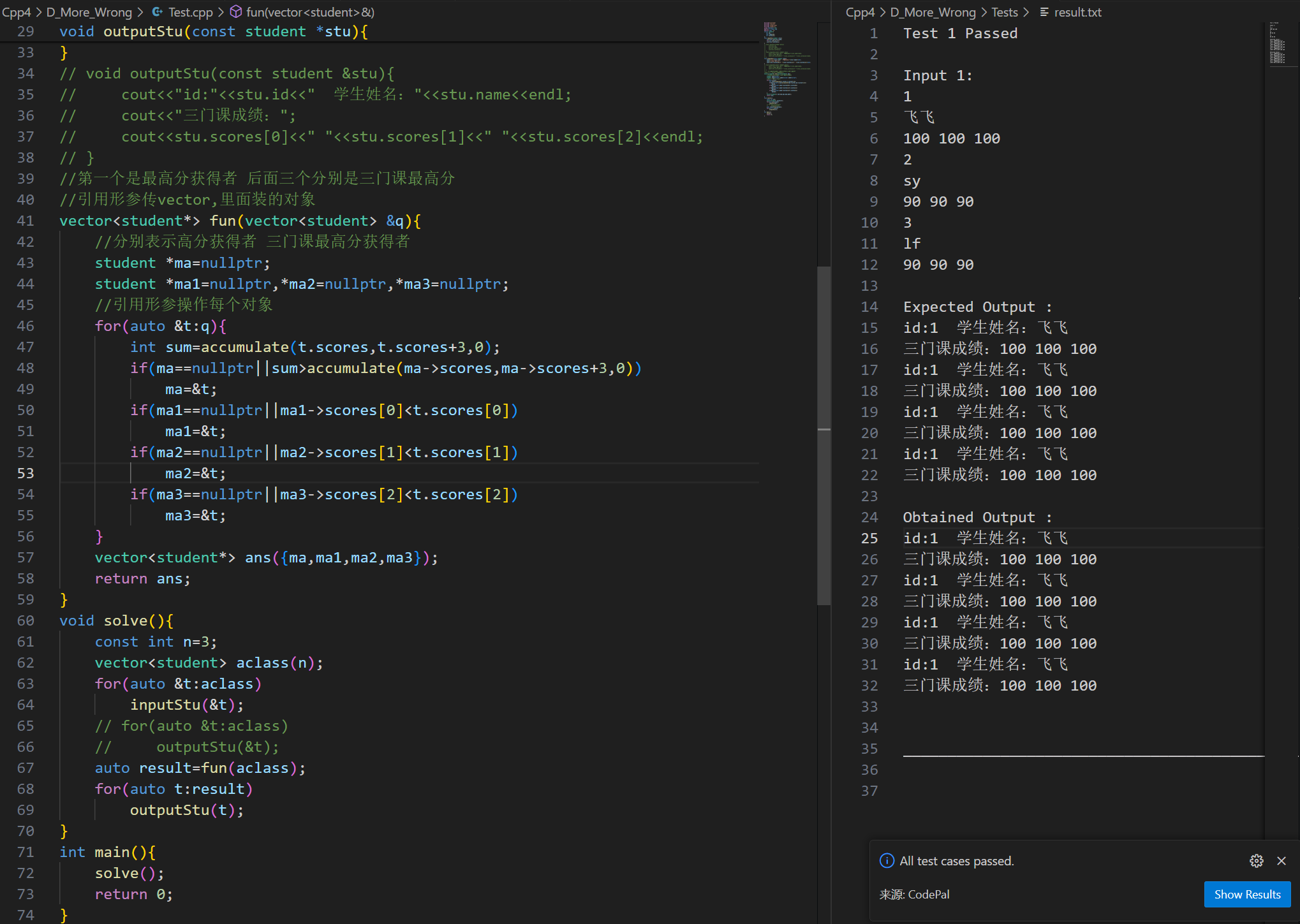
            ma3=&t;

    }

    vector<student\*> ans({ma,ma1,ma2,ma3});

    return ans;

}

****

**思考：inputStu和outputStu哪个形参设计中使用const更佳？为什么？请举例说明。**

**outputStu的形参设计中使用const更佳，因为输出值的函数使用const可以避免形参被改变，而输入值的函数有时候可能需要修改输入的参数。**

**❸重做课堂例题：修改一个整型数组中的原最大值为INT\_MAX。（20分）**

**主函数框架：**

**int main()**

**{ const int size=10;**

**int a[size], i;**

**for(i=0;i<size;i++) cin>>a[size];**

**// 声明变量 x //返回值设计为对应空间的值还是下标？或者对应空间的指针?**

**x=ReplaceMax(a,size);**

**a[x]=INT\_MAX; //设计为下标2-1**

**\*x=INT\_MAX; //设计为指针2-2**

**// ReplaceMax(a,size)=INT\_MAX; ??函数返回结果当左值 2-3**

**return 0;**

**}**

**要求按照3种设计给出完整的实现。体会引用做形参与其他类型参数的差异。**

* + **代码黏贴处：**

**ReplaceMax1: 返回值为待修改单元的下标**

int ReplaceMax1(int \*a,int size) {

    int i,max = 0;

    for(i=1;i<size;i++)

        if(a[max]<a[i])

            max = i;

    return max;

}

**ReplaceMax2: 返回待修改单元的地址**

int\* ReplaceMax2(int \*a,int size){

    int \*p=&a[0];

    for(int i=1;i<size;i++)

        if(a[i]>\*p){

            p=&a[i];

        }

    return p;

}

**ReplaceMax3: 返回待修改单元的引用**

int& ReplaceMax3(int \*a,int size){

    int j=0,max=a[0];

    for(int i=1;i<size;i++)

        if(a[i]>max){

            max = a[i];

            j=i;

        }

    return a[j];

}