**实验四 类和对象（二）**

**一、目的要求**

1. 理解面向对象的程序设计的特点；
2. 学会构造函数的使用和函数重载；
3. 学会对象指针的使用、对象的动态建立和释放；
4. 理解类的封装性和信息隐藏、析构函数的作用。
5. 理解构造和析构的调用时机和顺序；
6. 理解拷贝构造函数调用的时机，学会拷贝构造函数的使用。

**二、实验内容**

/\*student.h\*/

#include<iostream>

#include<string>

using namespace std;

class Student

{

private:

    string name,sex;

    int number;

    double score[2],avg;

public:

    Student();

    Student(int n,string na,string s,double s1,double s2):

           number(n), name(na), sex(s), score{s1,s2}{}

~Student();

    void input();      //输入信息函数

    void score\_avg();  //计算平均成绩函数

    double get\_s1();

    double get\_s2();

    double get\_avg();

    int get\_number();

    string get\_name();

    void show();       //显示信息函数

};

/\*member.cpp\*/

#include<iostream>

#include "student.h"

#include<string.h>

using namespace std;

Student::Student()

{

      number=0;

      score[2]={0};

      name="0";

      sex="0";

}

Student::~Student()

{

        cout<<"本程序由黄煜轩编写!"<<endl;

}

void Student::input()

{

        cout<<"学号：";

        cin>>number;

        cout<<"姓名：";

        cin>>name;

        cout<<"性别：";

        cin>>sex;

        cout<<"成绩1：";

        cin>>score[0];

        cout<<"成绩2：";

        cin>>score[1];

}

void Student::score\_avg()

{

        avg=(score[0]+score[1])/2;

}

double Student::get\_avg()

{

        return(avg);

}

double Student::get\_s1()

{

        return(score[0]);

}

double Student::get\_s2()

{

        return(score[1]);

}

int Student::get\_number()

{

        return(number);

}

string Student::get\_name()

{

        return(name);

}

void Student::show()

{

        cout<<"学号："<<number<<"  姓名："<<name

        <<"  性别："<<sex<<"  平均成绩："<<avg<<endl;

}

/\*file.cpp\*/

#include<iostream>

#include"student.h"

using namespace std;

int main()

{

    int n,i,j;

    cout<<"请输入学生人数n：";

    cin>>n;

    //下面进行对象的动态分配

    Student \*st= new Student[n];

    for(i=0;i<n;i++)

    {

        cout<<"请输入第"<<i+1<<"个学生的信息"<<endl;

        st[i].input();

    }

    //下面计算平均成绩

    double max1,max2;

    double min1,min2;

    double totle1,totle2,c\_avg1,c\_avg2;//全班总成绩和平均成绩

    //第一课程

    totle1=0.0;

    max1=st[0].get\_s1();

    min1=st[0].get\_s1();

    for(i=0;i<n;i++)

    {

        totle1 += st[i].get\_s1();

        if(st[i].get\_s1()>max1)

        {

            max1=st[i].get\_s1();

        }

         if(st[i].get\_s1()<min1)

        {

            min1=st[i].get\_s1();

        }

    }

    c\_avg1=totle1/n;

    //第二课程

    totle2=0.0;

    max2=st[0].get\_s2();

    min2=st[0].get\_s2();

    for(i=0;i<n;i++)

    {

        totle2 += st[i].get\_s2();

        if(st[i].get\_s2()>max2)

        {

            max2=st[i].get\_s2();

        }

         if(st[i].get\_s2()<min2)

        {

            min2=st[i].get\_s2();

        }

    }

    c\_avg2=totle2/n;

    //下面输出各课程的平均成绩、最高分、最低分

    cout<<"第一课程的平均成绩："<<c\_avg1<<" 最大值："<<max1<<" 最小值："<<min1<<endl;

    cout<<"第二课程的平均成绩："<<c\_avg2<<" 最大值："<<max2<<" 最小值："<<min2<<endl;

    //最后计算平均成绩并排序

    for(i=0;i<n;i++)

    {

        st[i].score\_avg();

    }

    //冒泡排序

    Student t;

    for(j=0;j<n-1;j++)//外侧比较轮数

       for(i=0;i<n-1-j;i++)//内侧每轮比较次数

       {

        if(st[i].get\_avg()<st[i+1].get\_avg())

        {

            t=st[i];

            st[i]=st[i+1];

            st[i+1]=t;

        }

       }

    //输出所有同学

    cout<<"下面按平均成绩从小到大输出："<<endl;

    for(i=0;i<n;i++)

    {

        st[i].show();

    }

    //查询

    int n0;

    cout<<"下面输入需要查询的学号查询(输入0来结束)："<<endl;

    cin>>n0;

    while(n0)

    {

        for(i=0;i<n;i++)

        {

        if(st[i].get\_number()==n0)

        {

            st[i].show();

        }

        }

        cout<<"下面输入需要查询的学号查询(输入0来结束)："<<endl;

        cin>>n0;

    }

    string name0;

    cout<<"下面输入需要查询的姓名查询(输入0来结束)："<<endl;

    cin>>name0;

    while(name0!="0")

    {

        for(i=0;i<n;i++)

        {

        if(st[i].get\_name()==name0)

        {

            st[i].show();

        }

        }

         cout<<"下面输入需要查询的姓名查询(输入0来结束)："<<endl;

         cin>>name0;

    }

    //最后释放动态内存

    delete[] st;

    return 0;

}

回答问题：

1. 先执行构造函数，待对象完成调用结束执行析构函数；
2. 1）使用一个已经创建完毕的对象来初始化一个新对象 2）值传递的方式给函数参数传值 3）以值方式返回局部对象
3. 不正确。析构函数是在对象销毁前执行的。
4. 防止形参指向的实参被改变。