

# 《面向对象程序课程设计》上机实验报告

班级： 2018 级 1 班      学号：   ====      姓名：====

A 部分：

一、实验题目

对如下多项式编写类定义：

$$a_0 + a_1x + a_2x^2 + \dots + a_nx^n$$

其中，n 为多项式的次数。完成如下功能：

- (1) 可存储任意大的多项式（提示：可用动态数组实现）。
- (2) 定义构造函数、析构函数、拷贝构造函数。
- (3) 包含一个 static 成员存储定义的多项式的数量。
- (4) 定义一个成员函数输出多项式。（可参照  $-x^4-6x^3+5$  格式输出）
- (5) 定义一个成员函数计算多项式的值。
- (6) 写 main 函数测试类的功能。
- (7) 采用多文件实现。

考虑：哪些成员函数可以声明为 const.

二、解决方案

定义一个 Duoxiangshi 类。用动态数组保存 n 个系数 an，用静态变量 number 保存多项式总个数。以上设置为保护，以保障安全。

定义了 Duoxiangshi();构造函数，~Duoxiangshi();析构函数，Duoxiangshi(const Duoxiangshi & temp);拷贝构造函数以实现深度拷贝，void set();设置函数，void print() const;打印函数，float calculate(float);计算函数。以上设置为公有以便调用。

使用多文件编写。

三、程序清单

```
//  
//  duoxiangshi.hpp  
//  20190611-第一次上机作业  
//  
//  Created by 荆煜 on 2019/6/14.  
//  Copyright © 2019 jingyu. All rights reserved.  
//
```

```
#ifndef duoxiangshi_hpp  
#define duoxiangshi_hpp  
#include <stdio.h>  
using namespace std;  
class Duoxiangshi  
{  
protected:  
    int n=1;  
    float *an=new float[n];
```

---

```

        static int number;
public:
    Duoxiangshi();
    ~Duoxiangshi();
    Duoxiangshi(const Duoxiangshi & temp);
    void set();
    void print() const;
    float calculate(float);
    static void print_number();
    static int return_number();
    friend Duoxiangshi operator+(const Duoxiangshi& a,const Duoxiangshi& b);
    Duoxiangshi operator-(const Duoxiangshi& temp);
    Duoxiangshi operator*(const Duoxiangshi& temp);
    Duoxiangshi& operator=(const Duoxiangshi& temp);
};
Duoxiangshi operator+(const Duoxiangshi& a,const Duoxiangshi& b);
#endif /* duoxiangshi_hpp */
//
//  duoxiangshi.cpp
//  20190611-第一次上机作业
//
//  Created by 荆煜 on 2019/6/14.
//  Copyright © 2019 jingyu. All rights reserved.
//

#include <iostream>
#include "duoxiangshi.hpp"
int Duoxiangshi::number=0;
Duoxiangshi::Duoxiangshi()
{
    number++;
}
void Duoxiangshi::set()
{
    do{
        if(n<=0)
            cout<<"n 应为正自然数"<<endl;
        cout<<"请输入"<<number<<"号多项式有多少项: ";
        cin>>n;
    }while(n<=0);
    for(int i=0;i<n;i++)
    {
        cout<<"请输入 a"<<i<<"=";
        cin>>an[i];
    }
}

```

---

```
    }
}
Duoxiangshi::~Duoxiangshi()
{
    number--;
}
Duoxiangshi::Duoxiangshi(const Duoxiangshi & temp)
{
    number++;
    n=temp.n;
    an=new float[n];
    for(int i=0;i<n;i++){
        an[i]=temp.an[i];
    }
}
void Duoxiangshi::print()const
{
    int panduan=0;
    for(int i=0;i<n;i++){
        switch(i)
        {
            case 0:
            {
                if(an[0]!=0){
                    panduan++;
                    cout<<an[0];
                }
                break;
            }
            case 1:
            {
                if(an[1]>0){
                    if(panduan>0)
                        cout<<" ";
                    if(an[1]!=1)
                        cout<<an[1]<<"X";
                    else
                        cout<<"X";
                    panduan++;
                }
                if(an[1]<0){
                    if(an[1]!=-1)
                        cout<<an[1]<<"X";
```

---

```
        else
            cout<<"-"<<"X";
            panduan++;
        }
        break;
    }
    default:
    {
        if(an[i]>0){
            if(panduan>0)
                cout<<"+";
            if(an[i]!=1)
                cout<<an[i]<<"X^"<<i;
            else
                cout<<"X^"<<i;
            panduan++;
        }
        if(an[i]<0){
            if(an[i]!=-1)
                cout<<an[i]<<"X^"<<i;
            else
                cout<<"-"<<"X^"<<i;
            panduan++;
        }
    }
}

}

}

}

float Duoxiangshi::calculate(float x)
{
    float sum=0,temp=1;
    for(int i=0;i<n;i++)
    {
        temp=an[i];
        for(int j=0;j<i;j++)
        {
            temp*=x;
        }
        sum+=temp;
    }
    return sum;
}

void Duoxiangshi::print_number()
{
```

---

```
        cout<<"共有"<<number<<"个多项式"<<endl;
    }
    int Duoxiangshi::return_number()
    {
        return number;
    }
    Duoxiangshi operator+(const Duoxiangshi& a,const Duoxiangshi& b)
    {
        Duoxiangshi duoxiangshi;
        if(a.n>b.n)
        {
            duoxiangshi.n=a.n;
            for(int i=0;i<b.n;i++)
            {
                duoxiangshi.an[i]=a.an[i]+b.an[i];
            }
            for(int j=b.n;j<a.n;j++)
            {
                duoxiangshi.an[j]=a.an[j];
            }
        }
        else
        {
            duoxiangshi.n=b.n;
            for(int i=0;i<a.n;i++)
            {
                duoxiangshi.an[i]=a.an[i]+b.an[i];
            }
            for(int j=a.n;j<b.n;j++)
            {
                duoxiangshi.an[j]=b.an[j];
            }
        }
        return duoxiangshi;
    }
    Duoxiangshi Duoxiangshi::operator-(const Duoxiangshi& temp)
    {
        Duoxiangshi duoxiangshi;
        if((this->n)>temp.n)
        {
            duoxiangshi.n=(this->n);
            for(int i=0;i<temp.n;i++)
            {
                duoxiangshi.an[i]=this->an[i]-temp.an[i];
```

---

```

        }
        for(int j=temp.n;j<this->n;j++)
        {
            duoxiangshi.an[j]=this->an[j];
        }
    }
    else
    {
        duoxiangshi.n=temp.n;
        for(int i=0;i<this->n;i++)
        {
            duoxiangshi.an[i]=this->an[i]-temp.an[i];
        }
        for(int j=this->n;j<temp.n;j++)
        {
            duoxiangshi.an[j]=(-temp.an[j]);
        }
    }
    return duoxiangshi;
}

Duoxiangshi Duoxiangshi::operator*(const Duoxiangshi& temp)
{
    Duoxiangshi duoxiangshi;
    duoxiangshi.n=(this->n)+temp.n;
    for(int i=0;i<duoxiangshi.n;i++)
        duoxiangshi.an[i]=0;
    for(int i=0;i<this->n;i++){
        for(int j=0;j<temp.n;j++){
            duoxiangshi.an[(i+j)]+=((this->an[i])*temp.an[j]);
        }
    }
    return duoxiangshi;
}

Duoxiangshi& Duoxiangshi::operator=(const Duoxiangshi &temp)
{
    this->n=temp.n;
    this->an=new float[n];
    for(int i=0;i<(this->n);i++)
    {
        this->an[i]=temp.an[i];
    }
    return *this;
}
//

```

---

```
// main.cpp
// 20190614-第二次上机作业
//
// Created by 荆煜 on 2019/6/14.
// Copyright © 2019 jingyu. All rights reserved.
//

#include <iostream>
#include "duoxiangshi.hpp"
int main(int argc, const char * argv[]) {
    float x;
    Duoxiangshi a1;
    a1.set();
    cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
    a1.print();
    cout<<endl<<endl<<"测试运算: "<<endl<<"请输入 x=";
    cin>>x;
    a1.print();
    cout<<"="<<a1.calculate(x)<<endl<<endl;
    Duoxiangshi a2;
    a2.set();
    cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
    a2.print();
    cout<<endl<<endl<<"测试\"+"": "<<endl;
    Duoxiangshi a3=a1+a2;
    cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
    a3.print();
    cout<<endl;
    Duoxiangshi::print_number();
    cout<<endl<<"测试\"-\"": "<<endl;
    Duoxiangshi a4=a1-a2;
    cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
    a4.print();
    cout<<endl;
    Duoxiangshi::print_number();
    cout<<endl<<"测试\"*\"": "<<endl;
    Duoxiangshi a5=a1*a2;
    cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
    a5.print();
    cout<<endl;
    Duoxiangshi::print_number();
    cout<<endl<<"测试\"=\"": "<<endl;
    Duoxiangshi a6;
    a6=a1;
```

```

cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
a6.print();
cout<<endl;
Duoxiangshi::print_number();
cout<<endl<<"测试拷贝构造函数"<<endl;
Duoxiangshi a7(a1);
cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
a7.print();
cout<<endl;
Duoxiangshi::print_number();
return 0;
}

```

#### 四、程序运行结果

```

20190614-第二次上机作业 My Mac Finished running 20190614-第二次上机作业 : 20190614-第二次上机作业
20190614-第二次上机作业 M
duoxiangshi.hpp A
duoxiangshi.cpp A
20190614-第二次上机作业 M
main.cpp M
Products

请输入1号多项式有多少项: 3
请输入a0=1
请输入a1=-1
请输入a2=3
多项式1号为: 1-X+3X^2
测试结果:
请输入x=2
1-X+3X^2=11
请输入2号多项式有多少项: 4
请输入a0=1
请输入a1=0
请输入a2=-5
请输入a3=7
多项式2号为: 1-5X^2+7X^3
测试结果:
多项式3号为: 2-X-2X^2+7X^3
共有3个多项式
测试结果:
多项式4号为: -X+8X^2-7X^3
共有4个多项式
测试结果:
多项式5号为: 1-X-2X^2+12X^3-22X^4+21X^5
共有5个多项式
测试结果:
多项式6号为: 1-X+3X^2
共有6个多项式
测试拷贝构造函数
多项式7号为: 1-X+3X^2
共有7个多项式
Program ended with exit code: 0

```



```
20190614-第二次上机作业 - My Mac
Finished running 20190614-第二次上机作业 : 20190614-第二次上机作业

20190614-第二次上机作业 M
duoxiangshi.hpp A
duoxiangshi.cpp A
20190614-第二次上机作业 M
main.cpp M
Products

请输入1号多项式有多少项: 3
请输入a0=2
请输入a1=2
请输入a2=2
多项式1号为: 2+2X+2X^2

测试运算:
请输入b0=1
2+2X+2X^2+6

请输入2号多项式有多少项: 3
请输入b0=4
请输入a1=5
请输入a2=6
多项式2号为: 4+5X+6X^2

测试"+":
多项式3号为: 6+7X+8X^2
共有3个多项式

测试"-":
多项式4号为: -2-3X-4X^2
共有4个多项式

测试"*":
多项式5号为: 8+18X+38X^2+22X^3+12X^4
共有5个多项式

测试"/":
多项式6号为: 2+2X+2X^2
共有6个多项式

测试成员构造函数
多项式7号为: 2+2X+2X^2
共有7个多项式
Program ended with exit code: 0
```

## 五、体会与总结

在拷贝构造函数中，会产生 `Duoxiangshi` 类型的临时对象，调用结束后执行析构函数使多项式数量 `number` 多减了一次，因此应注意+1。在 `print()`函数中，应考虑 0，负数等情况已满足实际显示需要。注意动态数组的个数 `n` 和静态变量必须初始化。

注：本学期共五个实验题目，实验报告分为五部分，分别用 A、B、C、D、E 描述。

# 《面向对象程序设计》上机实验报告

班级： 2018 级 1 班

学号： 55180123

姓名： 荆煜

B 部分：

六、实验题目

对如下多项式编写类定义：

$$a_0 + a_1x + a_2x^2 + \dots + a_nx^n$$

其中，n 为多项式的次数。完成如下功能：

- (8) 可存储任意大的多项式（提示：可用动态数组实现）。
- (9) 定义构造函数、析构函数、拷贝构造函数。
- (10) 包含一个 static 成员存储定义的多项式的数量。
- (11) 定义一个成员函数输出多项式。（可参照  $-x^4-6x^3+5$  格式输出）
- (12) 定义一个成员函数计算多项式的值。
- (13) 写 main 函数测试类的功能。
- (14) 采用多文件实现。
- (15) 重载 “+” 运算符，实现两个多项式相加。
- (16) 重载 “-” 运算符，实现两个多项式相减。
- (17) 重载 “\*” 运算符，实现两个多项式相乘。
- (18) 重载 “=” 运算符，实现两个多项式的赋值运算。

考虑：把其中某个运算符重载为友元函数。

七、解决方案

在实验一的基础上，增加了四个运算符的重载。其中，加号使用友元在类外实现。等号作为赋值运算符应与双目运算符做出区分。

八、程序清单

```
//  
// duoxiangshi.hpp  
// 20190611-第一次上机作业  
//  
// Created by 荆煜 on 2019/6/14.  
// Copyright © 2019 jingyu. All rights reserved.  
//
```

```
#ifndef duoxiangshi_hpp  
#define duoxiangshi_hpp  
#include <stdio.h>  
using namespace std;  
class Duoxiangshi  
{  
protected:  
    int n=1;
```

---

```
    float *an=new float[n];
    static int number;
public:
    Duoxiangshi();
    ~Duoxiangshi();
    Duoxiangshi(const Duoxiangshi & temp);
    void set();
    void print() const;
    float calculate(float);
    static void print_number();
    static int return_number();
    friend Duoxiangshi operator+(const Duoxiangshi& a,const Duoxiangshi& b);
    Duoxiangshi operator-(const Duoxiangshi& temp);
    Duoxiangshi operator*(const Duoxiangshi& temp);
    Duoxiangshi& operator=(const Duoxiangshi& temp);
};
Duoxiangshi operator+(const Duoxiangshi& a,const Duoxiangshi& b);
#endif /* duoxiangshi_hpp */
//
//  duoxiangshi.cpp
//  20190611-第一次上机作业
//
//  Created by 荆煜 on 2019/6/14.
//  Copyright © 2019 jingyu. All rights reserved.
//

#include <iostream>
#include "duoxiangshi.hpp"
int Duoxiangshi::number=0;
Duoxiangshi::Duoxiangshi()
{
    number++;
}
void Duoxiangshi::set()
{
    do{
        if(n<=0)
            cout<<"n 应为正自然数"<<endl;
        cout<<"请输入"<<number<<"号多项式有多少项: ";
        cin>>n;
    }while(n<=0);
    for(int i=0;i<n;i++)
    {
        cout<<"请输入 a"<<i<<"=";
```

---

```
        cin>>an[i];
    }
}
Duoxiangshi::~Duoxiangshi()
{
    number--;
}
Duoxiangshi::Duoxiangshi(const Duoxiangshi & temp)
{
    number++;
    n=temp.n;
    an=new float[n];
    for(int i=0;i<n;i++)
    {
        an[i]=temp.an[i];
    }
}
void Duoxiangshi::print()const
{
    int panduan=0;
    for(int i=0;i<n;i++){
        switch(i)
        {
            case 0:
            {
                if(an[0]!=0){
                    panduan++;
                    cout<<an[0];
                }
                break;
            }
            case 1:
            {
                if(an[1]>0){
                    if(panduan>0)
                        cout<<" ";
                    if(an[1]!=1)
                        cout<<an[1]<<"X";
                    else
                        cout<<"X";
                    panduan++;
                }
                if(an[1]<0){
                    if(an[1]!=-1)
```

```

        cout<<an[1]<<"X";
    else
        cout<<"-"<<"X";
    panduan++;
}
break;
}
default:
{
    if(an[i]>0){
        if(panduan>0)
            cout<<"+";
        if(an[i]!=1)
            cout<<an[i]<<"X^"<<i;
        else
            cout<<"X^"<<i;
        panduan++;
    }
    if(an[i]<0){
        if(an[i]!=-1)
            cout<<an[i]<<"X^"<<i;
        else
            cout<<"-"<<"X^"<<i;
        panduan++;
    }
}
}
}
}

float Duoxiangshi::calculate(float x)
{
    float sum=0,temp=1;
    for(int i=0;i<n;i++)
    {
        temp=an[i];
        for(int j=0;j<i;j++)
        {
            temp*=x;
        }
        sum+=temp;
    }
    return sum;
}

void Duoxiangshi::print_number()

```

---

```
{
    cout<<"共有"<<number<<"个多项式"<<endl;
}
int Duoxiangshi::return_number()
{
    return number;
}
Duoxiangshi operator+(const Duoxiangshi& a,const Duoxiangshi& b)
{
    Duoxiangshi duoxiangshi;
    if(a.n>b.n)
    {
        duoxiangshi.n=a.n;
        for(int i=0;i<b.n;i++)
        {
            duoxiangshi.an[i]=a.an[i]+b.an[i];
        }
        for(int j=b.n;j<a.n;j++)
        {
            duoxiangshi.an[j]=a.an[j];
        }
    }
    else
    {
        duoxiangshi.n=b.n;
        for(int i=0;i<a.n;i++)
        {
            duoxiangshi.an[i]=a.an[i]+b.an[i];
        }
        for(int j=a.n;j<b.n;j++)
        {
            duoxiangshi.an[j]=b.an[j];
        }
    }
    return duoxiangshi;
}
Duoxiangshi Duoxiangshi::operator-(const Duoxiangshi& temp)
{
    Duoxiangshi duoxiangshi;
    if((this->n)>temp.n)
    {
        duoxiangshi.n=(this->n);
        for(int i=0;i<temp.n;i++)
        {
```

---

```

        duoxiangshi.an[i]=this->an[i]-temp.an[i];
    }
    for(int j=temp.n;j<this->n;j++)
    {
        duoxiangshi.an[j]=this->an[j];
    }
}
else
{
    duoxiangshi.n=temp.n;
    for(int i=0;i<this->n;i++)
    {
        duoxiangshi.an[i]=this->an[i]-temp.an[i];
    }
    for(int j=this->n;j<temp.n;j++)
    {
        duoxiangshi.an[j]=(-temp.an[j]);
    }
}
return duoxiangshi;
}
Duoxiangshi Duoxiangshi::operator*(const Duoxiangshi& temp)
{
    Duoxiangshi duoxiangshi;
    duoxiangshi.n=(this->n)+temp.n;
    for(int i=0;i<duoxiangshi.n;i++)
        duoxiangshi.an[i]=0;
    for(int i=0;i<this->n;i++){
        for(int j=0;j<temp.n;j++){
            duoxiangshi.an[(i+j)]+=((this->an[i])*temp.an[j]);
        }
    }
    return duoxiangshi;
}
Duoxiangshi& Duoxiangshi::operator=(const Duoxiangshi &temp)
{
    this->n=temp.n;
    this->an=new float[n];
    for(int i=0;i<(bthis->n);i++)
    {
        this->an[i]=temp.an[i];
    }
    return *this;
}

```

---

```
//
//  main.cpp
//  20190614- 第二次上机作业
//
//  Created by 荆煜 on 2019/6/14.
//  Copyright © 2019 jingyu. All rights reserved.
//

#include <iostream>
#include "duoxiangshi.hpp"
int main(int argc, const char * argv[]) {
    float x;
    Duoxiangshi a1;
    a1.set();
    cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
    a1.print();
    cout<<endl<<endl<<"测试运算: "<<endl<<"请输入 x=";
    cin>>x;
    a1.print();
    cout<<"="<<a1.calculate(x)<<endl<<endl;
    Duoxiangshi a2;
    a2.set();
    cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
    a2.print();
    cout<<endl<<endl<<"测试\"+"": "<<endl;
    Duoxiangshi a3=a1+a2;
    cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
    a3.print();
    cout<<endl;
    Duoxiangshi::print_number();
    cout<<endl<<"测试\"-\": "<<endl;
    Duoxiangshi a4=a1-a2;
    cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
    a4.print();
    cout<<endl;
    Duoxiangshi::print_number();
    cout<<endl<<"测试\"*\": "<<endl;
    Duoxiangshi a5=a1*a2;
    cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
    a5.print();
    cout<<endl;
    Duoxiangshi::print_number();
    cout<<endl<<"测试\"=\": "<<endl;
    Duoxiangshi a6;
```



```

a6=a1;
cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
a6.print();
cout<<endl;
Duoxiangshi::print_number();
cout<<endl<<"测试拷贝构造函数"<<endl;
Duoxiangshi a7(a1);
cout<<"多项式"<<Duoxiangshi::return_number()<<"号为: ";
a7.print();
cout<<endl;
Duoxiangshi::print_number();
return 0;
}

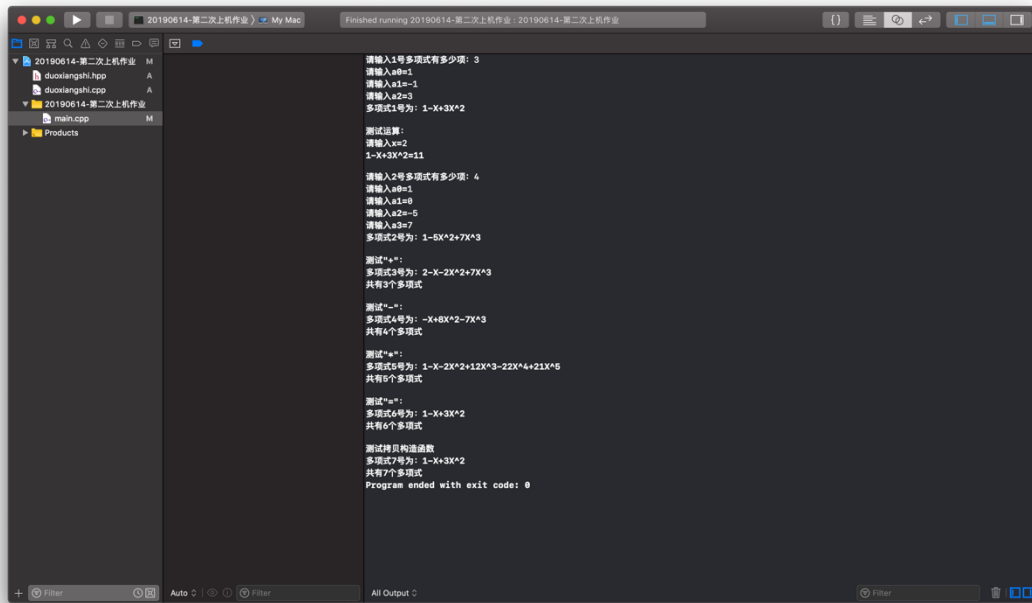
```

## 九、程序运行结果

```

20190614-第二次上机作业
请输入1号多项式有多少项: 3
请输入a0=2
请输入a1=2
请输入a2=2
多项式1号为: 2+2X+2X^2
测试运算:
请输入x=1
2+2X+2X^2=6
请输入2号多项式有多少项: 3
请输入a0=4
请输入a1=5
请输入a2=6
多项式2号为: 4+5X+6X^2
测试*=:
多项式3号为: 6+7X+8X^2
共有3个多项式
测试*=:
多项式4号为: -2-3X-4X^2
共有4个多项式
测试*=:
多项式5号为: 8+18X+38X^2+22X^3+12X^4
共有5个多项式
测试*=:
多项式6号为: 2+2X+2X^2
共有6个多项式
测试拷贝构造函数
多项式7号为: 2+2X+2X^2
共有7个多项式
Program ended with exit code: 0

```



```
20190614-第二次上机作业 - My Mac
Finished running 20190614-第二次上机作业 : 20190614-第二次上机作业

20190614-第二次上机作业 M
duoliangshi.hpp A
duoliangshi.cpp A
main.cpp M
Products

请输入1号多项式有多少项: 3
请输入a0=1
请输入a1=-1
请输入a2=3
多项式1号为: 1-X+3X^2

测试运算:
请输入s=2
1-X+3X^2=11

请输入2号多项式有多少项: 4
请输入a0=1
请输入a1=0
请输入a2=-5
请输入a3=7
多项式2号为: 1-5X^2+7X^3

测试"*=":
多项式3号为: 2-X-2X^2+7X^3
共有3个多项式

测试"-=":
多项式4号为: -X+6X^2-7X^3
共有4个多项式

测试"*=":
多项式5号为: 1-X-2X^2+12X^3-22X^4+21X^5
共有5个多项式

测试"-=":
多项式6号为: 1-X+3X^2
共有6个多项式

测试输出构造函数
多项式7号为: 1-X+3X^2
共有7个多项式
Program ended with exit code: 0
```

## 十、体会与总结

重载双目运算符在类外定义时，应有两个参数，在类内定义时，只需一个参数。赋值运算符应格外注意。

注：本学期共五个实验题目，实验报告分为五部分，分别用 A、B、C、D、E 描述。

---

## 《面向对象程序设计》上机实验报告

班级： 2018 级 1 班

学号： 55180123

姓名： 荆煜

C 部分：

### 十一、实验题目

C++的一般编译器都定义和封装了字符串功能，请模仿定义 `string` 类的实现，可以实现并支持如下功能：

- (1) `string s = "吉林大学";`
- (2) `string t = s;`
- (3) `string m; m = t;`
- (4) `m.length()` 函数测量字符串的长度
- (5) `m.cat(string const &)`连接字符串

### 十二、解决方案

将 `string` 作为 `char` 数组进行存储。通过对 `char` 数组的操作来完成对 `string` 的操作。

### 十三、程序清单

```
//  
// String.hpp  
// 20190614-第三次上机作业  
//  
// Created by 荆煜 on 2019/6/20.  
// Copyright © 2019 jingyu. All rights reserved.  
//
```

```
#ifndef String_hpp
```

```
#define String_hpp
```

```
#include <stdio.h>
```

```
#include <iostream>
```

```
class String
```

```
{
```

```
private:
```

```
    char* str;
```

```
    int len;
```

```
public:
```

```
    String ();
```

```
    ~String();
```

```
    String (const char*);
```

```
    String (const String&);
```

```
    String operator =(const char *);
```

```
    String operator =(const String&);
```

```
    int length();
```

```
    void print();
```

---

```
        String cat(const String& s);
};
#endif /* String_hpp */
//
//  String.cpp
//  20190614-第三次上机作业
//
//  Created by 荆煜 on 2019/6/20.
//  Copyright © 2019 jingyu. All rights reserved.
//
```

```
#include "String.hpp"
using namespace std;
String::String()
{
    len=0;
    str=new char[0];
}
String::~String()
{
    delete [] str;
}
String::String(const char* strlit)
{
    int i = 0;
    while(strlit[i]!='\0')
    {
        i++;
    }
    len=i;
    str=new char[len+1];
    for(int j=0;j<i;++j)
    {
        str[j]=strlit[j];
    }
}
String::String(const String& strbig)
{
    len=strbig.len;
    str=new char[len+1];
    for(int j=0;j<len;++j)
    {
        str[j]=strbig.str[j];
    }
}
```

---

```
}
String String::operator =(const char* ch)
{
    int i=0;
    while(ch[i]!='\0')
    {
        i++;
    }
    delete [] this->str;
    this->len=i;
    this->str=new char[len+1];
    for(int j=0;j<i;++j)
    {
        this->str[j]=ch[j];
    }
    return *this;
}

String String::operator =(const String& str1)
{
    this->len=str1.len;
    delete []this->str;
    str=new char[len+1];
    for(int i=0;i<len;++i)
    {
        this->str[i]=str1.str[i];
    }
    return *this;
}

int String::length()
{
    return this->len;
}

void String::print()
{
    for(int i=0;i<this->len;++i)
    {
        cout<<this->str[i];
    }
    cout<<endl;
}

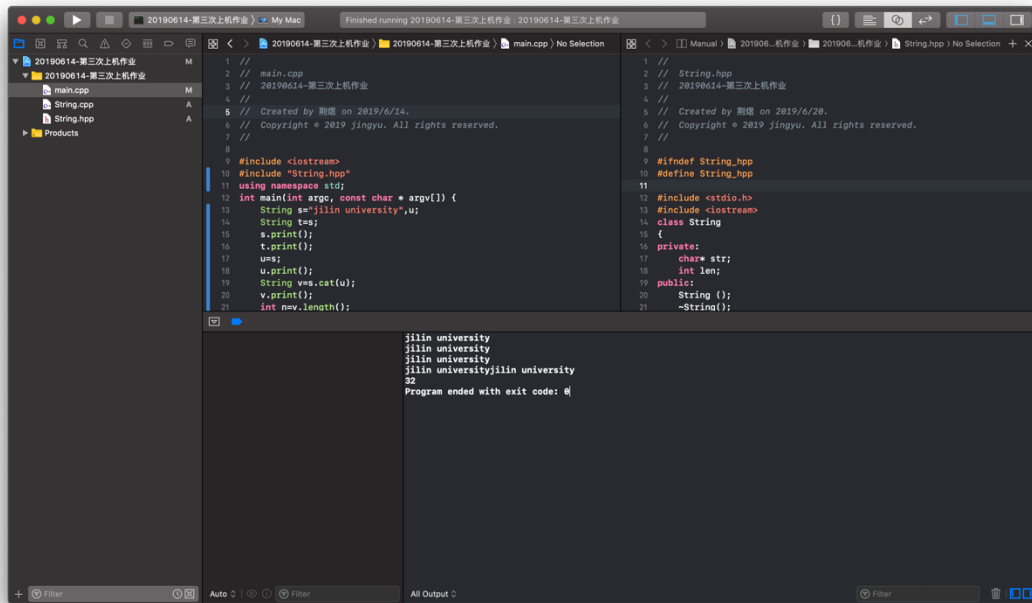
String String::cat(const String& s)
{
    String t;
    t.len=this->len+s.len;
```

---

```
t.str=new char[t.len+1];
for(int i=0;i<this->len;++i)
{
    t.str[i]=this->str[i];
}
for(int i=this->len;i<t.len;++i)
{
    t.str[i]=s.str[i-this->len];
}
return t;
}
//
//  main.cpp
//  20190614-第三次上机作业
//
//  Created by 荆煜 on 2019/6/14.
//  Copyright © 2019 jingyu. All rights reserved.
//
```

```
#include <iostream>
#include "String.hpp"
using namespace std;
int main(int argc, const char * argv[]) {
    String s="jilin university",u;
    String t=s;
    s.print();
    t.print();
    u=s;
    u.print();
    String v=s.cat(u);
    v.print();
    int n=v.length();
    cout<<n<<endl;
    return 0;
}
```

十四、程序运行结果



## 十五、体会与总结

char 数组中有\0 应加以注意。在重载构造函数中，若有多个参数不为空的构造函数，在开辟新的类空间且不符合自己定义的构造函数的参数时，不会自动执行默认构造函数。应注意增加一条 `String (){};`。在编写时，应注意引用的使用。

注：本学期共五个实验题目，实验报告分为五部分，分别用 A、B、C、D、E 描述。

---

## 《面向对象程序课程设计》上机实验报告

班级： 2018 级 1 班      学号： 55180123      姓名： 荆煜

D 部分：

### 十六、实验题目

我公司为仪器生产企业，目前生产摄像机和行车记录仪两种产品，分别销售给用户。

摄像机包含摄像、图像质量设定、编码算法等属性。

将摄像机增加相应芯片（具有操作菜单、自动拍摄、车速传感器、源代码等功能）后，形成一个行车记录仪。

要求：

设计摄像机类，并请根据下列不同的功能要求，采用不同的继承方式，设计行车记录仪类，并添加测试代码，体验不同继承方式下的成员访问属性。（类设计时可根据需要自行添加数据成员和其他成员函数。）

（1） 行车记录仪的芯片可以使用摄像机的摄像、图像质量设定功能。

    行车记录仪用户可以操作行车记录仪的操作菜单和摄像机的摄像功能。

（2） 行车记录仪的芯片可以使用摄像机的拍摄、图像质量设定功能。

    行车记录仪用户仅仅可以操作行车记录仪的操作菜单。

（3） 行车记录仪的芯片可以使用摄像机的拍摄、图像质量设定功能。

    行车记录仪用户仅仅可以操作行车记录仪的操作菜单

    同时其他公司购买行车记录仪，因该公司也用于销售，不得泄露其全部内容

课后：

（1） 采用组合方式设计行车记录仪类，增加相应测试代码，体验继承和组合的关系。

（2） 分别为继承和组合方式下为各类添加构造函数、析构函数，增加相应测试代码，体验对象的初始化和构造顺序。

（3） 将摄像机类和行车记录仪类功能相近的函数（如拍摄、编码等功能函数）设为同名函数，增加相应测试代码，体验同名函数覆盖。

（4） 为我公司建立一个多态的产品类层次结构，使用抽象类，测试时，创建一个基类指针的容器，通过基类指针调用虚函数，体验多态。

### 十七、解决方案

为更直观的体现继承和组合的关系和区别，分别使用继承和组合编写了两个程序。

前三个问题主要考察 `public`，`protected`，`private` 三种继承方式。

课后题（2）考察继承时的构造顺序和空间分配。

在基类中的某一函数，若为普通函数且在派生类中重新定义，则会发生 `override`。若为虚函数且参数相同，则可通过基类指针根据不同的类实现不同功能实现多态。若为虚函数且参数不同，依然会发生覆盖。此时可以使用 `using namespace + 类型名` 实现对基类中该函数的调用。

在菱形（多重）继承中，若发生派生类对基类变量的调用冲突时，可以使用虚拟继承加以解决。

### 十八、程序清单

// 以下为继承程序

// company.hpp



---

```
// 20190615-第四次上机作业
//
// Created by 荆煜 on 2019/6/20.
// Copyright © 2019 jingyu. All rights reserved.
//

#ifndef company_hpp
#define company_hpp

#include <stdio.h>
#include <iostream>
using namespace std;
class virtual_test{
public:
    virtual void name()=0;
};
class virtual_test_camera:public virtual_test{
public:
    void name(){
        cout<<"我是相机"<<endl;
    }
};
class virtual_test_camera_sony:public virtual_test_camera{
public:
    void name(){
        cout<<"我是索尼相机"<<endl;
    }
};
class virtual_test_camera_canon:public virtual_test_camera{
public:
    void name(){
        cout<<"我是佳能相机"<<endl;
    }
};
class Camera{
    void algorithmo(){
        cout<<"算法已展示"<<endl;
    }
protected:
    void photo_quality_set(){
        cout<<"图像质量设置功能正常"<<endl;
    }
    void data(){
        cout<<"可展示重要信息"<<endl;
    }
};
```

---

```
    }
public:
    Camera(){
        cout<<"Camera 已构造"<<endl;
    }
    ~Camera(){
        cout<<"Camera 已析构"<<endl;
    }
    void video(){
        cout<<"摄像功能正常"<<endl;
    }
};

class Chip_public:public Camera{
public:
    Chip_public(){
        cout<<"Chip_public 已构造"<<endl;

    }
    ~Chip_public(){
        cout<<"Chip_public 已析构"<<endl;

    }
    void menu(){
        cout<<"已打开菜单"<<endl;
    }
    void test(){
        cout<<endl<<"芯片功能测试: "<<endl;
        photo_quality_set();
        video();
        data();
        cout<<"测试结束"<<endl;
    }
};

class Chip_protected:protected Camera{
public:
    Chip_protected(){
        cout<<"Chip_protected 已构造"<<endl;
    }
    ~Chip_protected(){
        cout<<"Chip_protected 已析构"<<endl;
    }
    void menu(){
        cout<<"已打开菜单"<<endl;
    }
}
```

---

```

    void test(){
        cout<<endl<<"芯片功能测试: "<<endl;
        photo_quality_set();
        video();
        data();
        cout<<"测试结束"<<endl;
    }
};

class Chip_private:private Camera{
public:
    Chip_private(){
        cout<<"Chip_private 已构造"<<endl;
    }
    ~Chip_private(){
        cout<<"Chip_private 已析构"<<endl;
    }
    void menu(){
        cout<<"已打开菜单"<<endl;
    }
    void test(){
        cout<<endl<<"芯片功能测试: "<<endl;
        photo_quality_set();
        video();
        data();
        cout<<"测试结束"<<endl;
    }
};

class Override_test:public Chip_public{
public:
    void video(){
        cout<<"已优化摄像功能"<<endl;
    }
};

#endif /* company_hpp */
//
//  main.cpp
//  20190615-第四次上机作业
//
//  Created by 荆煜 on 2019/6/14.
//  Copyright © 2019 jingyu. All rights reserved.
//

#include <iostream>
#include "company.hpp"

```

---

```
using namespace std;
int main(int argc, const char * argv[]) {
    Chip_public chip_public;
    chip_public.test();
    cout<<endl<<"用户功能测试: "<<endl;
    chip_public.menu();
    chip_public.video();
    cout<<"测试结束"<<endl;
    cout<<endl;
    Chip_protected chip_protected;
    chip_protected.test();
    cout<<endl<<"用户功能测试: "<<endl;
    chip_protected.menu();
    //    chip_protected.video();
    cout<<"测试结束"<<endl;
    cout<<endl;
    Chip_private chip_private;
    chip_private.test();
    cout<<endl<<"用户功能测试: "<<endl;
    chip_private.menu();
    //    chip_private.video();
    cout<<"测试结束"<<endl<<endl<<"覆盖测试: "<<endl;
    Override_test override_test;
    chip_public.video();
    override_test.video();
    cout<<endl<<endl<<"多态测试: "<<endl;
    virtual_test *vt[3];
    vt[0]=new virtual_test_camera;
    vt[1]=new virtual_test_camera_canon;
    vt[2]=new virtual_test_camera_sony;
    for(int i=0;i<3;i++){
        vt[i]->name();
    }
    return 0;
}
```

*//以下为组合程序*

*// main.cpp*

*// 20190621-第四次上机作业*

*//*

*// Created by 荆煜 on 2019/6/21.*

*// Copyright © 2019 jingyu. All rights reserved.*

*//*

---

```
#include <iostream>
using namespace std;
class camera{
public:
    void cinematography();
    void setting_of_picture_resolution();
    void coding_algorithm();
};

void camera::cinematography()
{
    cout<<"Begin shooting."<<endl;
}

void camera::setting_of_picture_resolution()
{
    cout<<"Successfully set."<<endl;
}

void camera::coding_algorithm()
{
    cout<<"Successfully code."<<endl;
}

class chip
{
public:
    void menu();
    void AutoShoot();
    void SpeedSensor();
    void TheSourceCode();
    friend class camera;           //在 chip 类内将 camera 类声明为友元，即可使用
                                   camera 类里的函数
};

void chip::AutoShoot()
{
    cout<<"Autoshoot is good."<<endl;
}

void chip::menu()
{
    cout<<"There is menu."<<endl;
```

---

```
}

void chip::SpeedSensor()
{
    cout<<"Speed sensor is ready."<<endl;
}

void chip::TheSourceCode()
{
    cout<<"There is the source code."<<endl;
}

class AutomobileDateRecorder1{
private:
    camera c1;
    chip c2;           //camera 类和 chip 类在 AutomobileDateRecord1 类中是组合关系
public:
    void cinematography()
    {
        c1.cinematography();
    }
    void menu()
    {
        c2.menu();
        {
            c1.cinematography();
            c1.setting_of_picture_resolution();
        }
    }
};

class AutomobileDataRecorder2{
private:
    camera c1;
    chip c2;           //camera 类和 chip 类在 AutomobileDateRecord1 类中是组合关系
public:
    void menu()
    {
        c2.menu();
        {
            c1.cinematography();
            c1.setting_of_picture_resolution();
        }
    }
}
```

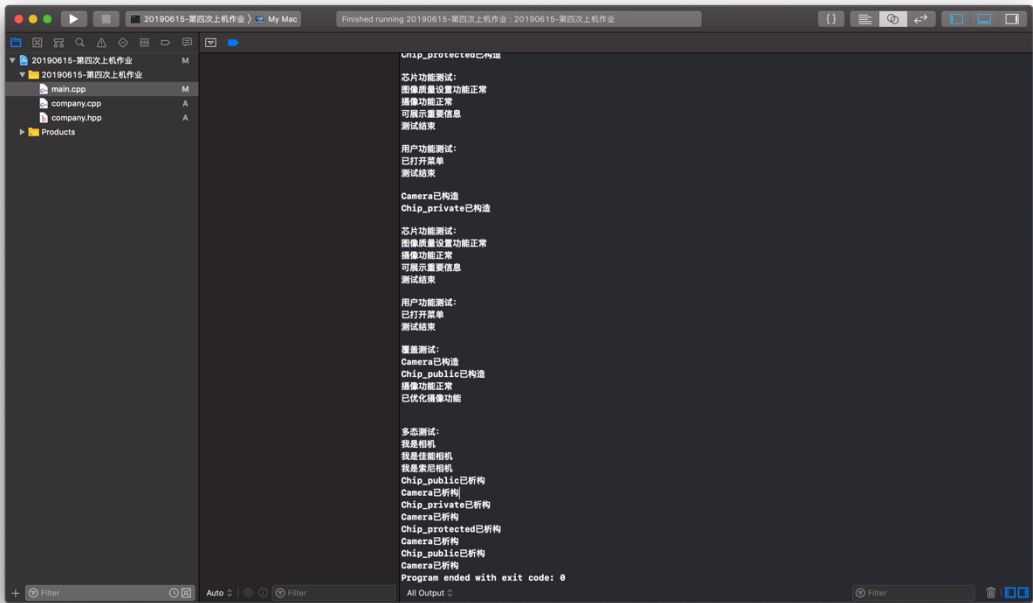
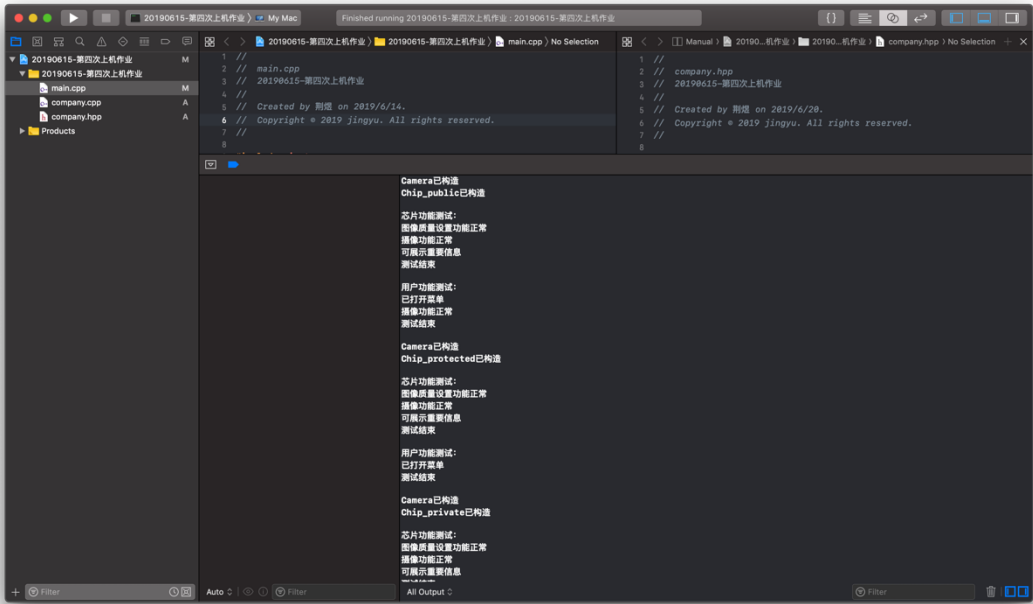
---

```
};
```

```
class AutomobileDataRecorder3{
private:
    camera c1;
    chip c2;           //camera 类和 chip 类在 AutomobileDateRecord1 类中是组合关系
public:
    void menu()
    {
        c2.menu();
        {
            c1.cinematography();
            c1.setting_of_picture_resolution();
        }
    }
};
```

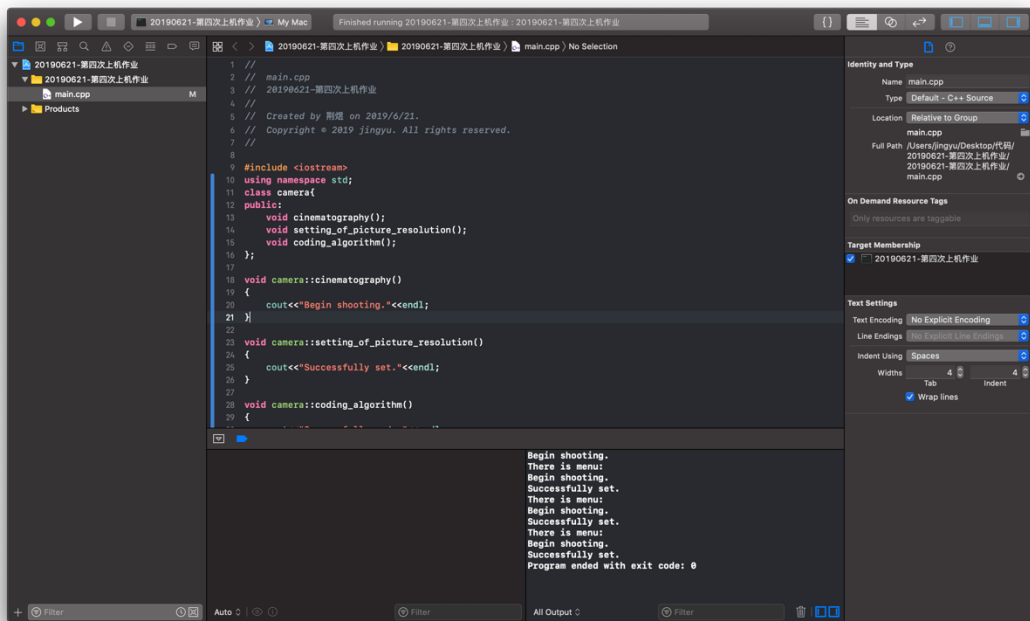
```
int main()
{
    AutomobileDateRecorder1 a1;
    AutomobileDataRecorder2 a2;
    AutomobileDataRecorder3 a3;
    a1.cinematography();
    a1.menu();
    a2.menu();
    a3.menu();
}
```

十九、程序运行结果  
以下为继承程序



以下为组合程序





## 二十、体会与总结

与组合相比，继承能使程序更有条理。在编写时发现，在基类中的某一函数，若为普通函数且在派生类中重新定义，则会发生 **override**。若为虚函数且参数相同，则可通过基类指针根据不同的类实现不同功能实现多态。若为虚函数且参数不同，依然会发生覆盖。此时可以使用 **using namespace + 类型名** 实现对基类中该函数的调用。在棱形（多重）继承中，若发生派生类对基类变量的调用冲突时，可以使用虚拟继承加以解决。

注：本学期共五个实验题目，实验报告分为五部分，分别用 A、B、C、D、E 描述。

---

## 《面向对象程序设计》上机实验报告

班级： 2018 级 1 班

学号： 55180123

姓名： 荆煜

E 部分：

二十一、实验题目

以下为定义的图形继承体系：

```
class Shape{
    public:
    // ...
    virtual double area(){}
};
class Circle:public Shape{
    public:
    double area(){...}
    // ...
};
class Triangle:public Shape{
    public:
    double area(){...}
    // ...
};
// ...
```

以下为容器类：

```
class Manage{
    shape  *a[100];
    public:
    // ...
}
```

要求完成 Manage 类的拷贝构造函数，实现深度拷贝；可以改动所有的类，并给出 main 函数测试程序。

二十二、解决方案

默认拷贝构造函数是浅拷贝，对于指针指示简单的将新指针指向了原空间，并不能实现空间及空间内内容的拷贝，很容易造成错误。因此在构造深度拷贝时应注意空间和内容的拷贝。在 main 测试程序中，我先将 100 个 shape 指针指向的 shape, circle, triangle 赋值，在将保存 shape 指针的 manage 类拷贝，通过断点检查指针是否指向新空间，在通过打印新的 manage 对象中 shape 指针指向三种对象的数据，来检查数据是否成功拷贝。

二十三、程序清单

```
//
// shape.hpp
// 20190615-第五次上机作业
//
// Created by 荆煜 on 2019/6/21.
```

---

```
// Copyright © 2019 jingyu. All rights reserved.  
//
```

```
#ifndef shape_hpp  
#define shape_hpp  
#include <iostream>  
#include <stdio.h>  
class Shape{  
public:  
    //...  
    int a;  
    virtual double area();  
    virtual Shape* new_shape(){  
        Shape* temp=new Shape(*this);  
        return temp;  
    }  
};  
class Circle:public Shape{  
public:  
    double area();  
    Shape * new_shape(){  
        Shape* temp=new Circle(*this);  
        return temp;  
    }  
    //...  
};  
class Triangle:public Shape{  
public:  
    double area();  
    Shape * new_shape(){  
        Shape* temp=new Triangle(*this);  
        return temp;  
    }  
    //...  
};  
//...  
class Manage{  
public:  
    Shape  *a[100];  
    Manage(){}  
    Manage(const Manage & temp);  
    void print(){  
        for(int i=0;i<100;i++){  
            a[i]->area();  
        }  
    }  
};
```

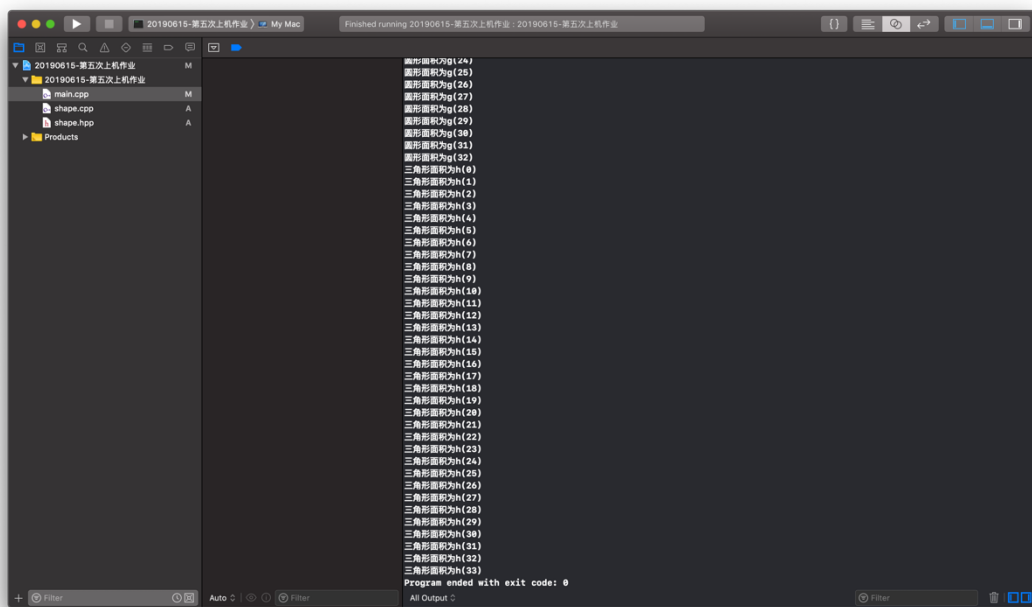
---

```
    }
}
//...
};

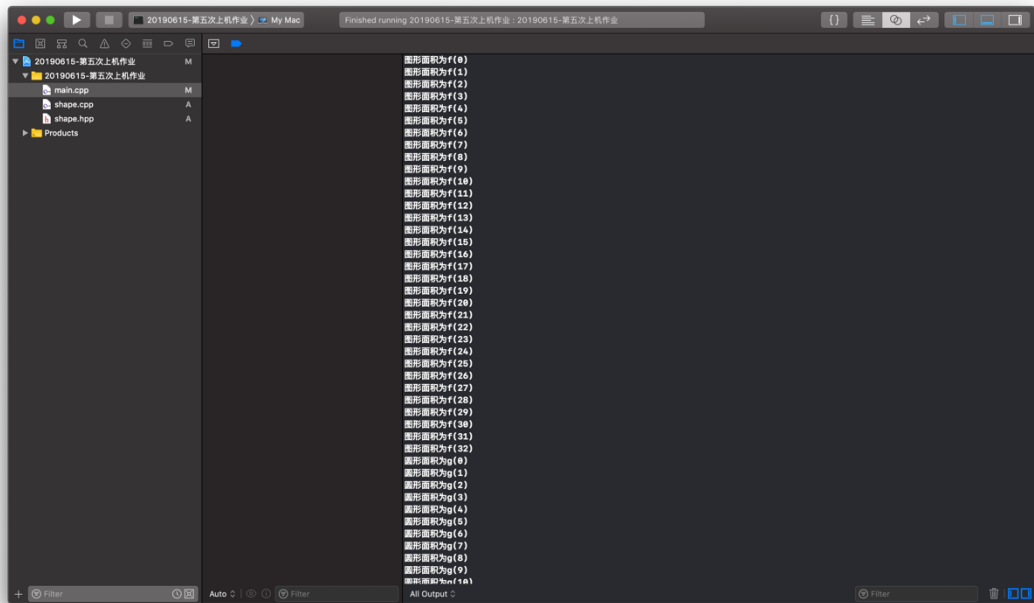
#ifdef /* shape_hpp */
//
//  shape.cpp
//  20190615-第五次上机作业
//
//  Created by 荆煜 on 2019/6/21.
//  Copyright © 2019 jingyu. All rights reserved.
//
#include "shape.hpp"
using namespace std;
double Shape::area(){
    cout<<"图形面积为 f("<<a<<")<<endl;
    return a;
}
double Circle::area(){
    cout<<"圆形面积为 g("<<a<<")<<endl;
    return a;
}
double Triangle::area(){
    cout<<"三角形面积为 h("<<a<<")<<endl;
    return a;
}
Manage::Manage(const Manage & temp){
    for(int i=0;i<100;i++){
        a[i]=temp.a[i]->new_shape();
    }
}
//
//  main.cpp
//  20190615-第五次上机作业
//
//  Created by 荆煜 on 2019/6/14.
//  Copyright © 2019 jingyu. All rights reserved.
//

#include <iostream>
#include "shape.hpp"
int main(int argc, const char * argv[]) {
    Manage manage1;
```

```
Shape shape[33];
for(int i=0;i<33;i++){
    shape[i].a=i;
    manage1.a[i]=&shape[i];
}
Circle circle[33];
for(int i=0;i<33;i++){
    circle[i].a=i;
    manage1.a[i+33]=&circle[i];
}
Triangle triangle[34];
for(int i=0;i<34;i++){
    triangle[i].a=i;
    manage1.a[i+66]=&triangle[i];
}
manage1.print();
std::cout<<std::endl<<std::endl;
Manage manage2=manage1;
manage2.print();
return 0;
}
```







## 二十五、体会与总结

深度拷贝应注意指针指向的空间问题。应先分配新的空间，再将原对象的数据逐一拷贝到新空间中。

注：本学期共五个实验题目，实验报告分为五部分，分别用 A、B、C、D、E 描述。