程 序 设 计 A 实 验 指 导

实验UNIT 04 类与对象

《程序设计》课程组

武 汉 大 学 计 算 机 学 院

第4讲上机实验

实验目的：

1. 类的声明和使用
2. 类的声明和对象的声明
3. 具有不同访问属性的成员的访问方式
4. 观察构造函数和析构函数的执行过程
5. 学习类的组合使用方法
6. Debug功能观察程序流程，跟踪观察类的构造函数、析构函数、成员函数的执行顺序

第4讲上机实验

实验任务：

1. 课堂练习：类和对象，重点构造函数和析构函数

2. 编程练习：类的声明和实现

第4讲上机实验

- 实验步骤提示：

1. 为每个题目建立一个新的控制台项目文件；
2. 向其中提交一个类声明的头文件，一个包含类实现的代码C++源文件；
3. 向其中提交测试类的主函数及其代码；
4. 录入代码，检查是否有错误？有则改之；
5. 选择菜单“生成解决方案”编译源程序；
6. 执行程序，观察输出结果是否正确观察输出结果是否正确，如果有错误，可以执行第6步；
7. 使用debug功能：跟踪观察类的构造函数、析构函数、成员函数的执行顺序

现在开始课堂练习！

练习内容：类和对象，

重点构造函数和析构函数

类的骨干

所有的类都有着一个类似的中枢骨干，外号“**Big Four**”：

构造函数＋析构函数＋拷贝构造函数＋赋值运算符重载

或者，通常称为“**Big Three**”：一个或多个构造函数＋一个析构函数＋一个拷贝赋值运算符。

练习1：构造函数初始值列表

```
// 成员初始化表的使用
#include <iostream>
using namespace std;
Class Sample
{
    private:
        int x;
        int &rx;
        const float pi;
    public:
        Sample(int x1):x(x1), pi(3.14), rx(x)
        { }
```

请回答两个问题：

（1）执行时，这里的初始化顺序是什么？

（2）pi和rx是否必须初始化？为什么？

练习1：构造函数初始值列表

```
        void Print()
        {
            cout<<"x="<<x<<" "<< <<"rx="<<rx<<" “
                 << "pi="<<pi<<endl;
        }
};
int main()
{
    Sample a(10);
    a.Print();
    return 0;
}
```

请分析程序的运行结果，并实际测试一下！

练习2：构造函数初始值列表

下面来看带有默认实参的构造函数：

```
// 日期类的定义。函数成员定义在类体中。
// date.h
#include <iostream>
using namespace std;
class Date                                  //定义日期类Date
{
  public:                                   //声明类成员
    Date(int y=2011, int m=1, int d=1);
    void ShowDate( )
  private:
    int year;     int month;     int day;
};                          //以括号及分号结束，体现封装
```

练习2：构造函数初始值列表

下面来看带有默认实参的构造函数：

```
// date.cpp
#include “date.h”

Date::Date(int y, int m, int d)
{
    year=y;  month=m;  day=d;
  cout<<"constructing…"<<endl;
}
void Date::ShowDate( )
{  cout<<"Date:  "<<year<<"."<<month<<"."<<day;
    cout<<endl;
}
```

练习2：构造函数初始值列表

下面来看带有默认实参的构造函数：

```
//  main.cpp
#include “date.h”
int main()
{
    Date date1;
    Date date2(2005);
    Date date3(2006,12,15);
    cout<<"date1:";      date1.ShowDate();
    cout<<"date2:";      date2.ShowDate();
    cout<<"date3:";      date3.ShowDate();
    return 0;
}
```

请按照注释说明的文件名，创建项目文件和源代码文件，然后分析程序的运行结果，并实际测试验证！

练习3：析构函数的定义

```
// 日期类的定义。
// date.h
#include <iostream>
using namespace std;
class Date                              //定义日期类Date
{
  public:                               //声明类成员
    Date(int y=2011, int m=1, int d=1);
    ~Date();
    void ShowDate( )
  private:
    int year;     int month;     int day;
};                  //以括号及分号结束，体现封装
```

练习3：析构函数的定义

```
// date.cpp
#include “date.h”
Date::Date(int y, int m, int d)
{
    year=y;  month=m;  day=d;
  cout<<"constructing…"<<endl;
}
void Date::ShowDate( )
{  cout<<"Date：  "<<year<<"."<<month<<"."<<day;
    cout<<endl;
}
Date::～Date()
{  cout<<"destructing…"<<endl;
}
```

练习3：　析构函数的定义

```
//  main.cpp
#include “date.h”
int main()
{
    Date date1(1999,4,20);
    cout<<"date1:"<<endl;
    date1.ShowDate();
    Date date2(2004,10,15);
    cout<<"date2:"<<endl;
    date2.ShowDate();
    return 0;
}
```

请按照注释说明的文件名，创建项目文件和源代码文件，然后分析程序的运行结果，并实际测试验证！

练习4：构造函数和析构函数

请仔细分析接下来两页ppt的复数类程序，指出程序的运行结果，并实际测试验证！

```
// 析构函数和带默认参数的构造函数。
//  注意构造函数和析构函数的调用顺序。
#include <iostream>
using namespace std;
#include <math.h>
class Complex            //定义复数类
{ public:
        Complex(double r=0.0,double i=0.0);  //带默认参数构造函数的声明
        ~Complex();                          //析构函数的声明
        double abscomplex();                 //求复数的模
  private:
        double real;
        double imag;
};
Complex::Complex(double r,double i)   //构造函数的实现
{       cout<<"constructing..."<<endl;
        real=r;
        imag=i;
        cout<<"real:"<<real<<",imag:"<<imag<<endl;
}
```

```
Complex::~Complex()                          //析构函数的实现
{  cout<<"destructing...";
   cout<<"real:"<<real<<",imag:"<<imag <<endl;
}
double Complex::abscomplex()                 //成员函数的实现
{      double t;
        t=real*real+imag*imag;
        return sqrt(t);
}
int main()                          //主函数
{
Complex A(1.1,2.2),B=A;        //定义复数类对象A，自动调用构造函数
cout<<"abs of complex A="<<A.abscomplex()<<endl; //对象A调用成员函数
cout<<"abs of complex B="<<B.abscomplex()<<endl; //对象B调用成员函数
return 0;  //在程序结束前自动调用析构函数
}
```

Wuhan University

本次课堂练习结束！

上机编程练习任务

练习内容：类和对象的编程练习

第4讲上机任务

类和对象编程练习：

4-10、设计一个用于人事管理的“人员”类。由于考虑到通用性，这里只抽象出所有类型人员都具有的属性：编号、性别、出生日期、身份证号等。其中“出生日期”声明为一个内嵌子对象“日期”。用成员函数实现对人员信息的录入和显示。要求包括：构造函数和析构函数、复制 构造函数、内联成员函数、带默认形参值的成员函数、类的组合。

4-13、定义一个Circle类，有数据成员radius（半径），成员函数getArea()，计算圆的面积。构造一个Circle的对象进行测试。

第4讲上机任务

类和对象编程练习：

4-19、编写一个名为CPU的类，描述一个CPU以下的信息：时钟频率，最大不会超过3000MHz；字长可以是32位或64位；核数可以是单核、双核或四核；是否支持超线程。各项信息要求使用位域来表示。通过输出sizeof(CPU)来观察该类的字节数。

4-20、定义一个复数类Complex，使得下面的代码能够工作：

```
Complex c1(3,5);          //用复数3+5i初始化c1
Complex c2=4.5;           //用实数4.5初始化c2
c1.add(c2);               //将c1和3c2相加，结果保存在c1中
c1.show();                //将c1输出（这时的结果应该是7.5+5i）
```

第4讲上机任务

类和对象编程练习：

4-12、定义一个DataType（数据类型）类，能处理包括字符型、整型、浮点型三种数据，给出其构造函数。请自行完善主函数及其测试语句。

4-14、定义一个Tree（树）类，有采用ages（树龄），成员函数grow(int years)对ages加上years，age()显示Tree对象的ages值。

本 讲 结 束