# 实验七 构造函数和析构函数

## 一、 问题描述

## 

## 实验目的：

 掌握构造函数和析构函数 的若干基本概念和特性，并能够应用于程序编写。

 掌握验证性实验的基本方法和过程(认知、实验、总结）。

## 实验内容：

分别编写一段测试代码来回答任务书中的相关问题（每一个问题，用一个 工程文件，同时需要记录相应的调试过程），具体问题请参考“实验任务 说明07.doc”；

调试的过程；（动态调试的相关截图，比如 设置断点、查看当前变量值等）；

编译出来的可执行程序单独放在一个目录下（bin/exe/debug目录下，同时 附上输入数据说明和输出结果）

## 实验过程

1. **选择题**

1.1、定义一个类的对象时，系统自动调用（ **B** ）函数

A 成员函数 B 构造函数 C 析构函数 D 复制构造函数

1.2、下面对构造函数的不正确描述是（ **B** ）

A 系统可以提供默认的构造函数

B 构造函数可以有参数，所以也可以有返回值

C 构造函数可以重载

D 构造函数可以设置默认参数

1.3、在下面类的声明中，错误的语句是（**A**）

class Sample {

public:

Sample(int val); //A，类中变量的声明不能带初始化

~Sample(); //B

private:

int a=2.5; //C

};

1.4、假定Myclass是一个类，执行下列语句时，构造函数调用（**3**）次

Myclass a[3],\*p[2]; // 生成对象a[3]时会调用3次，定义指针\*p[2]不会调用构造函数

1. **程序阅读题，分析并讨论结果**

|  |  |  |  |
| --- | --- | --- | --- |
| 1 在分析结果的基础上，重点描述构造函数运行过程  备注：这是一个带参数的构造函数  #include <iostream>  using namespace std;  class A  {  private:  int x,y;  public:  A(int a,int b)  { x=a;  y=b;  }  void print(){cout<<”x=”<<x<<”,y=”<<y  <<endl;}  };  void main()  {  A a(-5,10);  a.print();  }  // **答：创建类类型的新对象，将-5和10传入构造函数，并将x和y赋值为-5、10。执行print()函数，打印x=-5，y=10。** | | 2 构造函数重载执行  备注：什么是函数重载？在运行时，如何识别重载的函数？构造函数重载的意义是什么？（请回顾：在标准库类型string中初始化的方式有几种？）  #include <iostream>  using namespace std;  class A  {  private:  int x,y;  public:  A() { x=0; y=1; }  A(int a) { x=a; y=10; }  A(int a,int b) { x=a; y=b; }  void print(){  cout<<”x=”<<x<<”,y=”<<y <<endl; }  };  void main()  { A a1,a2(5),a3(-5.-10);  a1.print();a2.print();a3.print();  }  **// 答：构造函数的重载：具有相同的函数名字,而参数的个数或参数的类型或位置不相同。** | |
| 3 根据结果，指出不足，分析原因，并尝试修改  备注：用户没有定义构造函数，系统默认构造函数会自动执行（看起来，似乎什么也不做）  #include <iostream>  using namespace std;  class Time {  private:  int hour,minute,second;  public:  void disp();  };  void Time::disp() {  cout<<hour<<”小时”  <<minute<<”分钟”  <<second<<”秒”<<endl;  }  void main()  {  Time time;  time.disp();  }  **// 答：用户没有定义构造函数，系统默认构造函数会自动执行，但是是无参的，没有对各个变量初始化，调用disp函数进行打印会存在风险。可以定义含有hour，minute，second三个int参数的构造函数用于初始化。** | 4 分析下列程序，给出结果  #include <iostream>  using namespace std;  class Date  { public:  **Date(int,int,int);**  Date(int,int);  Date(int);  Date();  void display();  private:  int month;  int day;  int year;  };  Date::Date(int m,int d,int y):  month(m),day(d),year(y) {}  Date::Date(int m,int d):  month(m),day(d) {year=2005;}  Date::Date(int m):  month(m){day=1;year=2005;}  Date::Date()  {month=1;day=1;year=2005;}  void Date::display()  { cout<<month<<”/”<<day<<”/”<<year<<endl;}  int main()  {  Date d1(10,13,2005);  Date d2(12,30);  Date d3(10);  Date d4;  d1.display();  d2.display();  d3.display();  d4.display();  return 0;  }  // **答：控制台将打印10 / 13 / 2005，12 / 30 / 2005**  **10 / 1 / 2005，1 / 1 / 2005。三参数的Date构造函数可以自定义年月日，两参数的Date构造函数可以自定义月日，一参数的Date构造函数可以自定义日，无参数的Date构造函数采用默认值。** | | |
| 5 将第四题中程序的**Date(int,int,int);** 修改为默认参数，即：  Date(int m=1,int d=1,int y=2005);  分析程序是否有问题，若有错误，给出出错提示，深究其原因。（即，能否在保留：Date(int m=1,int d=1,int y=2005); 的前提下，其输出结果和第四题的输出结果一致？）  **// 答：存在错误，创建d2,d3,d4对象时将会出现有多个构造函数的实例与参数列表匹配的报错。这是因为如果构造函数的全部参数都指定了默认值，则在定义对象时可以给一个或几个实参，也可以不给出实参。在一个类中定义了全部是默认参数的构造函数后，不能再定义重载构造函数。** |
| 6、从初始化列表角度，分析代码中可能出现的问题  #include <iostream>  using namespace std;  class Obj {  public:  Obj(int k): j(k),i(j) { }  void print(void) {  cout<<i<<endl<<j<<endl;  }  private:  int i;int j;  };  int main()  { Obj.obj(2);  Obj.print();  return 0;  }  **// 答：初始化列表中，由于将j的值给i做初始化，最后输出的i的值是不确定的。** | 7、改正下列程序中的错误，并说明理由 | | |
| 1 | | Include <iostream>;**using namespace std;** |
| 2 | | Class Student { |
| 3 | | **public:** |
| 4 | | Void Student( )**;** |
| 5 | | Void display ( ) |
| 6 | | { **cout**<<’no:’<<no<<endl |
| 7 | | cout<<name:<<name<<endl; |
| 8 | | cout<<score:<<score<<endl;} |
| 9 | | private: |
| 10 | | int no,char \*name,float score; |
| 11 | | } |

1. **简答题**
   1. 设计一个类时，往往会有两类成员函数，从效果上，都是使得对象有特定值。 请从函数的执行特性等角度分析他们的区别：

（1）构造函数及其重载

（2）用来进行初始值设置的成员函数：void setdata(int,int,int);

答：构造函数是一种特殊的成员函数，与其他成员函数不同，不需要用户来调用，而是在建立对象时自动执行。建立的对象是有名称的对象或是未命名的对象，显示定义的对象或隐式生成的对象。构造函数的自动执行是指无需通过对象调用就可以自动执行。构造函数的名字必须与类名同名，而不能由用户任意命名。它不具有任何类型，不返回任何值；

* 1. 构造函数书写时，有哪几种情形只能用初始化列表，而不能用函数体中对数据 成员赋值？请逐一给出代码说明。

答：初始化列表以一个冒号开始，接着是一个以逗号分隔的数据成员列表，每个数据成员后面跟一个放在圆括号中的初始化式；构造函数初始化式只在构造函数的定义中而不是声明中指定。很多时候，初始化列表可以转化为通过在函数体内对数据成员赋值来实现。但是，在某些情况下，只能使用初始化列表：

① 某个类中的类成员，没有默认构造函数。（若没有为类成员提供初始化式，则编译器会隐式地使用类成员的默认构造函数。而该类成员若恰好没有默认

构造函数，那么编译器尝试使用默认构造函数就会失败。② 某个类中有const成员。③ 某个类中有引用类型的成员；④ 如果类存在继承关系，派生类必须在其初始化列表中调用基类的构造函数。

* 1. 对象是现实世界事务的映射，类抽取了对象的共性。请分析，在类的设计过程 中，会有什么情况下，程序员会把数据成员定义成const？

答：任何不会修改数据成员都应该声明为const类型。如果在编写代码时，不慎修改了const数据成员，编译器将报错，会提高程序的健壮性。

**四、程序设计题**

4.1 针对1.4，编写代码实现

1. 设计思路

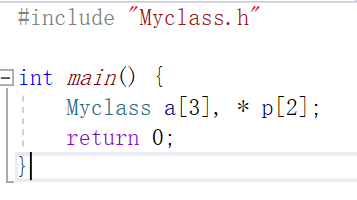
① 实验思路

构造函数一共调用了三次，在生成对象a[3]时会调用3次，定义指针\*p[2]不会调用构造函数。

② 设计Myclass.h头文件

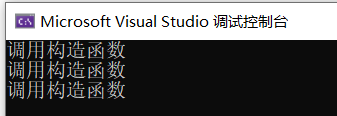


③设计Myclass.cpp文件



（2）实验结果：

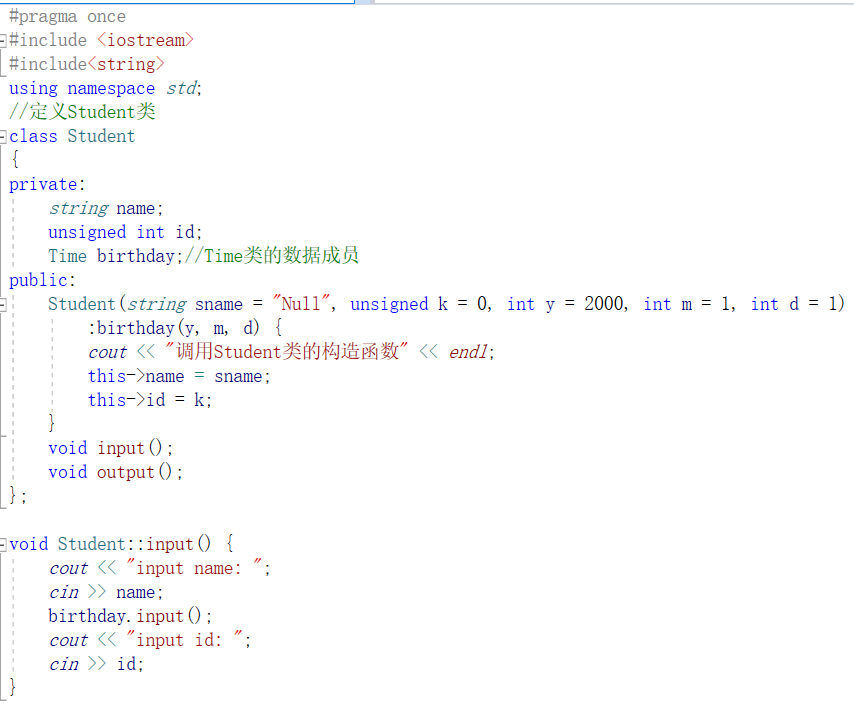
验证构造函数调用了三次



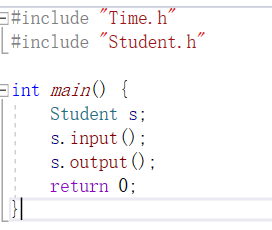
4.2 设计一个日期类Time，用来表示一个学生类Student中的学生的出生年月信 息（类成员）。编写代码，说明两个类的构造函数的执行顺序。

（1）Time.h头文件：



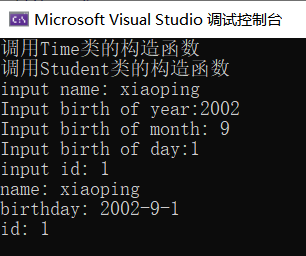


（2）Main.cpp文件：



（4）实验结果

执行代码，发现是先调用Time类的构造函数，再调用Student类的构造函数。



## 附录

源程序文件项目清单： 4.1 4.2