

C++ 程序设计

课程设计

刘振安 刘燕君 孙 忱 编著



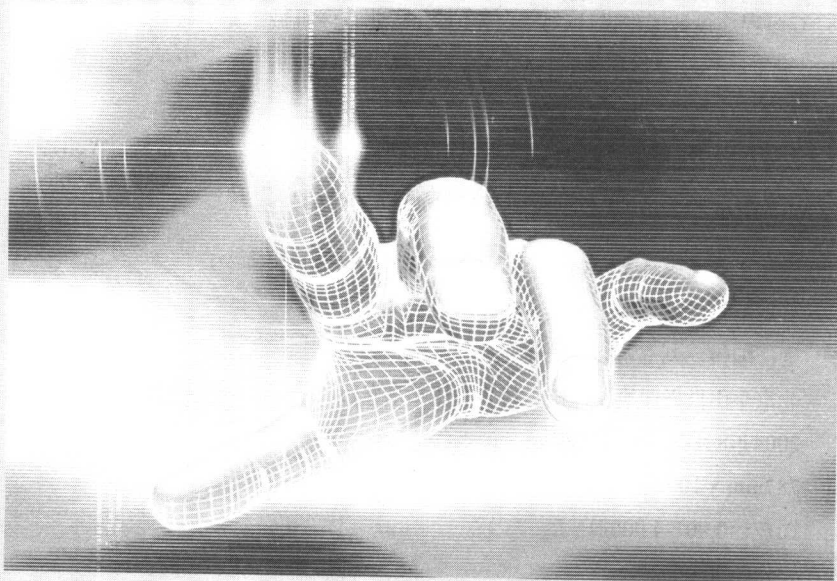
机械工业出版社
China Machine Press

高等院校计算机课程设计指导丛书

C++ 程序设计

课程设计

刘振安 刘燕君 孙 忱 编著



机械工业出版社
China Machine Press

课程设计可以充分弥补课堂教学和实验中知识深度和广度有限的问题,更好地帮助学生系统地掌握该门课程的主要内容。

本书是一本独立于具体的C++语言教科书的课程设计辅导,重点放在C++语言的基本特征上,结合实际应用,通过详细的实例,循序渐进地启发学生完成设计。书中给出的实例都很完整,并给出了测试样例,是一本很好的教学辅导参考书。

版权所有,侵权必究。

图书在版编目(CIP)数据

C++程序设计课程设计/刘振安等编著. —北京:机械工业出版社,2004.8
(高等院校计算机课程设计指导丛书)

ISBN 7-111-14565-8

I. C… II. 刘… III. C语言-程序设计-高等学校-教学参考资料 IV. TP312

中国版本图书馆CIP数据核字(2004)第057876号

机械工业出版社(北京市西城区百万庄大街22号 邮政编码 100037)

策划编辑:温莉芳

责任编辑:刘立卿

北京牛山世兴印刷厂印刷·新华书店北京发行所发行

2004年8月第1版第1次印刷

787mm×1092mm1/16·13.75印张

印数:0 001-4 000册

定价:20.00元

凡购本书,如有倒页、脱页、缺页,由本社发行部调换
本社购书热线:(010) 68326294

丛书序言

近年来,我国在计算机应用、计算机软件和电子类相关专业的人才培养方面,取得了长足的进展,每年的毕业生都有数十万人。但是这些毕业生走进企业、公司、政府机构或研究单位之后,往往深刻地感觉到缺乏实际开发设计项目的经验,不善于综合运用所学理论,对知识的把握缺乏融会贯通的能力。

综合考察目前高等院校教学大纲、课程设置以及内容安排等方面的情况,多数学校还是比较重视训练学生的实际设计能力。但是,从安排设计实践的内容上看,基本上是围绕相关课程教学内容而展开的,不能够构成对实际问题的解决方案;从配套程序的规模上看,一般只是几十行到几百行的源代码,或者是一个单独电路的设计,远远小于一个小型项目的规模;从设计的结构上看,由于设计实践是围绕着课程教学内容而进行的,问题已经高度抽象,学生很难得到有关综合运用所学知识的整体训练机会。而且,这些内容相对简单、问题域已经高度抽象、规模较小的设计实践一人基本上就能完成,学生几乎无法通过这些设计实践,去真正获得有关项目管理和团队协作等方面的基本训练和工作经验。

由此可以看出,大多数学校对学生实际设计能力的训练与国外知名大学和国内精品课程相比较,还是存在一些差距的。为此,机械工业出版社华章分社和一批高等院校的教师,针对当前高等院校计算机硬件、软件和电子类相关课程教学中存在的问题,参考国内外知名大学相关课程成功的教学经验,设计编写了这套“高等院校计算机课程设计指导丛书”,其目的就是通过课程设计的一系列训练,把知识获取和项目实践两个方面有机地结合起来。

在这套“高等院校计算机课程设计指导丛书”中的每一门课程设计里,都安排了由多个子项目组成的一个课程设计项目。学生们可以在教师的指导下,逐步设计实现这些子项目,并最终完成一个功能相对完整,可以运行的系统,其代码可以是数千行,甚至上万行。通过这种设计课程,学生一方面可以结合课程的教学内容循序渐进地进行设计方面的实践训练,另一方面,在参与一系列子项目的实践过程中,还能提高如何综合运用所学知识解决实际问题的能力,以及获得有关项目管理和团队合作等等众多方面的具体经验,增强对相关课程具体内容的理解和掌握能力,培养对整体课程知识综合运用和融会贯通能力。

参加丛书编写的各高等院校的教师都有着丰富的教学、科研,以及与企业合作开发项目等多方面的经验。每个课程设计中的子项目和整体项目,都来自教师们具体的科研和设计开发实践,所选设计项目与教学内容配合紧密,项目的难度与规模适宜。

最后,感谢机械工业出版社华章分社编辑们的大力支持,使出版有关这套丛书的计划,从单纯的构想演化成带有油墨芳香的真实。

丛书写作组

2004年7月

前言

语言课程应注重边学边练，但由于课堂教学和实验的深度和广度有限，练习的深度也受到一定限制。为了弥补这一点，特设计了本课程设计。

本课程设计的主要特点如下：

(1) 它独立于具体的C++语言教科书，重点放在C++语言的基本特征上，以“不变”应“万变”，涵盖C++语言的重要基础知识。

(2) 结合实际应用的要求，使课程设计既覆盖知识点，又接近工程实际需要。通过激发学习兴趣，调动学生主动学习的积极性，并引导他们根据实际编程要求，训练自己实际分析问题的能力，并养成良好的编程习惯。

(3) 通过详细的实例，循序渐进地启发学生完成设计。课程设计将要求、算法和源程序分开，为学生创造独立思考的条件。学生在充分理解要求和算法的前提下，完全可以不按书中提供的参考程序，设计自己的应用程序。

(4) 课程设计分为基本部分与扩展部分，以满足不同学校和学生要求。

(5) 提供综合课程设计，以期进一步锻炼学生使用面向对象方法思考问题及动手的能力。这些综合实验还可以供学生分工合作，以培养团队协作精神。

(6) 对同一类型的实验，提供不同的实现方法，以满足不同学校和学生的要求。

另外，在实际编程中，为了提高编程质量，对空行、空格和注释均有要求。本书也尽可能根据实际编程要求给出空行、空格和注释。有时因为标题和页码等实际原因，也会适当减少空行、空格和注释；但希望学生在书写代码时，还是应该严格按照要求处理，以便建立良好的编程风格。

全书共分11章。第1章是课程设计目标；第2章是使用类和对象设计回应程序；第3章是设计菜单选择趣味程序；第4章是通过组合和派生构成新的类；第5章是职工信息表；第6章是后缀表达式求值；第7章是求解有理数分式方程；第8章是字符串类；第9章是统计学生成绩；第10章是通信录；第11章是学生成绩管理程序。

本课程设计曾在一些班级试用，有些同学还在此基础上进行功能扩充，获得很好的效果。还有的学校将其用作毕业设计的参考资料，也取得可喜的成绩。为此，对他们表示感谢。

参加本书编写的还有刘燕君和孙忱等，他们放弃寒假的休息，不仅参与编写，还仔细调试程序，逐字逐句校对，为本书花费了大量心血。

刘振安

中国科学技术大学

2004年3月26日

目 录

丛书序言

前言

第1章 课程学习目标1

1.1 目的1

1.2 课程设计结构1

1.3 评价标准3

第2章 使用类和对象设计回应程序5

2.1 设计要求5

2.2 分析类的设计形式6

2.3 使用字符串设计类7

2.3.1 使用字符串设计类的总体考虑7

2.3.2 设计成员函数8

2.4 使用指针设计类9

2.4.1 设计类9

2.4.2 设计成员函数10

2.5 使用string类13

2.5.1 设计类及其头文件13

2.5.2 设计成员函数13

2.6 设计项目文件15

2.7 评价标准16

第3章 设计菜单选择趣味程序17

3.1 设计一个菜单程序17

3.1.1 设计要求17

3.1.2 设计思想18

3.2 设计菜单项20

3.2.1 菜单项设计要求20

3.2.2 求解方程21

3.2.3 使用类和数组求解出圈顺序23

3.2.4 使用类和向量求解出圈顺序25

3.3 项目文件结构27

3.4 程序清单29

3.5 程序运行示范37

3.6 评价标准39

第4章 通过组合和派生构成新的类41

4.1 使用包含设计的方法41

4.1.1 设计题目41

4.1.2 设计要求41

4.1.3 设计思想42

4.1.4 参考程序及运行结果43

4.1.5 分析46

4.2 使用模板的方法47

4.2.1 设计题目47

4.2.2 设计要求47

4.2.3 参考程序和运行结果48

4.3 使用继承的方法51

4.3.1 设计题目51

4.3.2 设计要求51

4.3.3 设计思想52

4.3.4 参考程序和运行结果54

4.4 使用模板继承的方法57

4.4.1 设计题目57

4.4.2 设计要求57

4.4.3 程序清单58

4.5 评价标准61

第5章 职工信息表63

5.1 使用类和数组的方法63

5.1.1 设计题目63

5.1.2 设计要求63

5.1.3 设计思想63

5.1.4 参考程序及运行结果65

5.1.5 评价标准	68	9.5 评价标准	137
5.2 使用类和向量的方法	69	第10章 通信录	139
5.2.1 设计界面	69	10.1 功能设计要求	139
5.2.2 设计重点	69	10.2 使用list类的成员函数	140
5.2.3 参考程序	69	10.3 结构和类	141
5.2.4 测试程序	74	10.3.1 结构Record	141
5.2.5 评价标准	77	10.3.2 类RecordList	142
第6章 后缀表达式求值	79	10.3.3 类AddressList	142
6.1 设计思想	79	10.3.4 类AddressBook	143
6.2 参考程序和运行实例	79	10.3.5 类的UML图	145
6.3 评价标准	84	10.4 参考程序	145
第7章 求解有理数分式方程	85	10.4.1 address文件	145
7.1 设计要求	85	10.4.2 AddressList文件	148
7.1.1 抽象Rational类	85	10.4.3 AddressBook文件	150
7.1.2 设计Rational类	86	10.5 测试示例	161
7.2 程序清单	89	10.5.1 菜单项及空表和空文件测试	161
7.3 求解有理数方程	97	10.5.2 测试建表和文件存储	163
7.4 评价标准	98	10.5.3 测试查询和删除记录	164
第8章 字符串类	99	10.6 评价标准	166
8.1 设计字符串类	99	第11章 学生成绩管理程序	167
8.1.1 抽象String类	99	11.1 设计要求	167
8.1.2 设计String类	100	11.1.1 功能设计要求	167
8.2 程序清单	104	11.1.2 总体设计	169
8.3 评价标准	113	11.2 参考程序	175
第9章 统计学生成绩	115	11.2.1 student文件	175
9.1 设计要求	115	11.2.2 StuInfoVec文件	178
9.2 设计思想	115	11.2.3 StuInfoManager文件	182
9.3 文件及函数组成	119	11.3 测试	201
9.4 参考程序	122	11.4 评价标准	210
		主要参考文献	211



第 1 章

课程设计目标

由于各校的情况不一,为了便于各校根据自身的特点和教学计划,选择相应的课程设计内容,本章简要介绍本课程的设计题目及其预期目标。

1.1 目的

一般来讲,课程设计比教学实验复杂一些,涉及的深度广些,并更加实用。目的是通过课程设计的综合训练,培养学生实际分析问题、编程和动手能力,最终目标是想通过课程设计的形式,帮助学生系统掌握该门课程的主要内容,更好地完成教学任务。另外,课程设计中的较大的综合设计,可以分成几个小项目供学生分工合作,以培养团队协作精神。

1.2 课程结构设计

结构化程序设计使用的是功能抽象,面向对象程序设计不仅能进行功能抽象,而且能进行数据抽象。“对象”实际上是功能抽象和数据抽象的统一。C++语言的“对象”是“类”的实例,程序设计的基础是设计类,所以类的有关概念都是重点,尤其要抓住抽象、封装、继承和多态性等要素。

类设计的重点是如何选择数据成员和成员函数。成员函数设计中的难点是选择函数类型及其参数传递方式。数据类型及程序控制方式仍然是C++语言的基础;数组、指针、类和结构的使用方法是编程的核心技术。学生学习时,常常避开多文件编程和使用文件,但这些都是程序设计员必备的知识,因此本课程设计特意加强了这方面的训练。

本课程设计共选择10个设计题目,不是用一个独立的例子涵盖这些知识点,而是按层次逐步深入。为了使学生理解它们之间如何相互配合,设计要求使用接近实际需要的方式编程。这些实验题目及其简要说明如下:

1. 使用类和对象设计回应程序(第2章)

设计类就是根据数据封装的要求,抽象出合适的类。换言之,就是如何选择数据成员和成员函数。本章的任务就是根据测试程序及其输出结果,设计一个满足要求的类。这个类涉及到类的构造函数和析构函数,并要求使用多种方法实现本课程设计。

2. 使用菜单选择趣味程序(第3章)

菜单是应用程序的界面。控制语句用来改变程序执行的顺序,是实现结构化程序设计的

基础。本章的任务是设计一个含有多个菜单项并且比较实用的菜单，通过菜单选择不同的功能。要求使用类实现这些菜单项，并熟悉多文件编程和面向对象程序设计思想。

3. 通过组合和派生构成新的类(第4章)

本章的设计任务是使用“Point”类产生“Line”类。这可以通过包含及派生两种方法实现，并要求分别给出使用模板实现的程序。目的是使学生掌握在不同的实现方法中，如何设计相应的构造函数和复制构造函数，进一步理解程序如何调用它们及析构函数的执行顺序。

公有继承的赋值兼容性规则是一个重要规则。通过这个设计，可以进一步理解这个概念，为理解虚函数的多态性打下基础。本章的设计实际上是4个题目，可以根据教学要求取舍。

4. 职工信息表(第5章)

本章的设计任务是设计职工信息表，并由此产生一个信息简表。这里使用数组，利用赋值兼容规则实现简表，并使用虚函数实现多态性，完成显示不同简表信息的任务。

本章还给出使用菜单和向量实现的设计方案，以便进一步理解向量的使用方法。这个要求可以作为选做项目，也可以根据教学需要加以取舍或增加新的要求。

5. 后缀表达式求值(第6章)

本章的设计任务是使用模板设计后缀表达式求值，模拟一个计算器操作。目的是进一步熟悉模板及模板之间的关联方法。设计本身并不难，但涉及堆栈的知识。可以增加测试要求，理解后缀表达式及堆栈的应用。

6. 求解有理数分式方程(第7章)

本章课程设计的任务是设计一个“有理数”类，用来求解有理数分式方程。这涉及到大量的运算符重载。

7. 字符串类(第8章)

本章课程设计的重点是设计一个“字符串”类。字符串数据抽象表示的对象是字符序列。字符串在许多与用户进行文本交互的应用系统里常常使用。字符串类的开发不仅说明了一个比较复杂的数据类型的建立过程，同时也介绍了C++语言的一些重要特征，帮助读者强化对一些概念的理解。

8. 统计学生成绩(第9章)

本章课程设计的重点是文件操作。这是一个非常有用而重要的技术，需要多多练习。编制实用程序都离不开文件存取，这方面应该给予足够的重视。

9. 通信录(第10章)

本章课程设计是设计一个实用的小型通信录程序，具有查询和删除功能，并且能够打开或修改指定文件及将多个文件组成一个文件。它完全使用类来实现，充分体现了面向对象的程序设计特点。

本章中的参考程序使用双向链表保存数据的底层数据结构，并使用C++标准库提供的list来实现这个链表的结构。由于利用list类，所以实现起来很简单。虽然这种方法超出一般

的教材，但只要学习一下标准库，还是容易理解的。C++推荐使用标准库，这也是目前编程的方向，所以本课程设计作为选做内容，可以根据实际教学情况决定，目的是使学生熟悉和了解C++的标准库。

10. 学生成绩管理程序(第11章)

本章课程设计是设计一个实用的小型学生成绩管理程序，它要求不用链表，而用向量来设计这个程序。它有查询和检索等功能，并且能够对指定文件操作，也可将多个文件组成一个文件。

建议将本课程设计列为必做内容，以便学生了解如何编制一个实用程序，从而进一步锻炼实际工作能力。也可以要求学生在熟悉本程序的基础上，改用数组来实现本设计。或者根据实际教学情况决定增删内容。

1.3 评价标准

因为书中已经有参考程序，所以一般情况下都能完成预定设计。如果只是按照程序去做，其分数只能在85分以下。为了证明学生已经掌握设计中所涵盖的知识点，应该向学生提一些问题，例如程序如何实现及其原理等。下列评价标准中对“全部正确”的定义，其实就含有这种面试手段。由于各校情况不一，所以就没有在各章的评价中规定必须提问题才算完成设计任务。

另外，程序的可读性均包含在正确性中记分，不再单列考核标准。

一般遵循如下规律评价：

- (1) 严格控制90分，其标准是有创意。
- (2) 85分以上，必须全部正确，并有一定改进或者能正确回答设计中的问题。
- (3) 有少许失误，可给75~83分。
- (4) 错误不多，给60~73分。
- (5) 没有完成特殊规定的要求，则不及格。

因为课程设计主要是锻炼学生，所以应该鼓励学生不要有负担，放手去做，激发他们钻研问题的兴趣。因此，应该尽力让学生均能通过，避免不及格现象。另外，评分标准也可以只设“通过”、“没通过”和“优秀”3档以激发学生的学习兴趣。



第 2 章

使用类和对象设计回应程序

设计类就是根据数据封装的要求，抽象出合适的类。换言之，就是如何选择数据成员和成员函数。成员函数也像函数一样，需要正确地选择函数类型及其参数传递方式。本章的任务就是根据测试程序及其输出结果，设计一个满足要求的类。

本章的设计还涉及到类的特殊函数：构造函数和析构函数。设计的实现方法并非惟一，本章将介绍3种典型的实现方法。

2.1 设计要求

1. 测试程序

假设类名为Welcome，测试程序如下：

```
void main()
{
    Welcome we;
    Welcome you(we);

    you.Display();
    you.Set("Thank you.");
    cout<<you.Get()<<endl;

    you.talk();
    you.Display();
}
```

2. 测试程序的输出结果

测试程序的输出结果如下：

```
Welcome!
Thank you.
输入: How about you?
输出: How about you?
输入: Fine, thank you.
输出: Fine, thank you.
输入: OK
All right!
```

```
Goodbye!  
OK  
Goodbye!  
Goodbye!
```

3. 对类的要求

要求按如下原则设计Welcome类:

- (1) 在头文件cpp1.h中声明类。
- (2) 在cpp1.cpp中实现类。
- (3) 使用内联函数定义Display()函数。
- (4) 至少使用两种不同的数据类型设计属性值。

2.2 分析类的设计形式

分析的依据是根据测试程序和输出结果,重点放在分析这个类应该具有哪些数据成员和成员函数才能满足要求上。至于如何实现成员函数,则放在后一步。

根据测试程序和输出结果研究类的组成,下面使用注释,对每条语句进行初步分析,以便给出可能的结构。

```
void main()  
{  
    Welcome we;                //使用默认参数的构造函数建立对象we  
    Welcome you(we);           //使用we的参数,通过复制构造函数建立对象you  
  
    you.Display();             //成员函数Display()用来显示属性  
    you.Set("Thank you.");     //成员函数Set()用来设置属性值  
    cout<<you.Get()<<endl;    //成员函数Get()用来得到属性值  
  
    you.talk();                //成员函数talk实现对话和回应  
    you.Display();  
}
```

从表面上看,很容易认为语句“Welcome we;”调用的是无参数构造函数。从第一个输出语句“you.Display();”的输出结果是“Welcome!”,可以推知该类具有一个参数的构造函数,而且使用默认参数,将属性赋值为字符串“Welcome!”。由此可见,这个类有一个构造函数和一个复制构造函数。

类有一个用来修改数据成员的Set()成员函数和取数据成员的Get()成员函数。成员函数Display()用来显示数据成员,而talk()成员函数则用来实现对话和回应。

从输出结果可以知道,它有析构函数。这样一来,它总共有7个成员函数和一个数据成员。类的设计关键是如何选择数据成员的数据类型以及构造函数的形式。

从第一条输出结果可知，它是由下面3条语句实现：

```
Welcome we;  
Welcome you(we);  
you.Display();
```

由此可见，对象使用默认参数初始化数据成员。析构函数的输出是：

```
Goodbye!
```

根据以上分析可知，最后的函数原型取决于数据成员。

从对话中可见，要求能够接受具有空格的字符串。数据成员为数组或者指针，都可以实现这一要求。另外，使用string类也可以实现。本设计就根据这3种情况分别实现之。

2.3 使用字符串设计类

首先使用字符数组设计类的属性，然后根据输出设计相应的成员函数。这是人们一般喜欢采取的办法，有人认为这是最简单易懂的一种。读者可以分析一下，到底哪种简单。

2.3.1 使用字符串设计类的总体考虑

类的声明如下：

```
//cpp2.h  
#if !defined(CPP1_H)  
#define CPP1_H  
  
#include <iostream>  
using namespace std;  
  
class Welcome  
{  
    private:  
        char str[128];  
  
    public:  
        Welcome(char s[]="Welcome!");//使用默认参数  
        Welcome(Welcome&);  
        void Set(char []);  
        void Display();  
        char *Get();  
        void talk();  
        ~Welcome();  
};
```

//因为是字符串数组，所以返回指针

```
#endif
```

构造函数使用默认参数时，必须在声明时设置默认参数，并且不能使用“;”号。

应要求头文件中使用条件编译方式，必须熟练掌握这种方式。

在cpp2.cpp文件中，别忘记使用如下包含语句：

```
#include "cpp2.h"
```

2.3.2 设计成员函数

1. 构造函数

两个构造函数的设计都很容易，在复制构造函数中，引用类的对象作为参数，假设这个引用对象为s，则s的数据成员是s.str。

对字符串进行赋值时，一般使用strcpy()函数。

```
Welcome::Welcome(char s[])
{
    strcpy(str,s);
}

Welcome::Welcome(Welcome&s)
{
    strcpy(str, s.str);
}
```

2. talk()函数

根据输出结果可以知道，这个成员函数里面是使用循环语句完成对话，而且结束条件是输入字符串“OK”。从它使用语句“you.Display();”输出OK，可以知道会话时的输入被赋给对象的数据成员。

从输出结果可以推知，析构函数被调用了3次，即主程序产生3个对象。从主程序中可以明显地看出它产生的两个对象we和you，因此另一个对象是在talk中产生。

从它只输出一个OK，而且是在析构一个对象之后产生的，可以肯定它不是临时对象。有关这个对象的信息为：

```
All right!
Goodbye!
```

显然这个对象是在结束会话之后产生的，对象的属性值为“All right!”。

通过以上分析，可以写出它的源程序如下：

```
//talk()成员函数
```

```
void Welcome::talk()
{
    for(;;)
    {
        cout<<"输入: ";
        gets(str);
        if(strcmp(str, "OK") ==0) break;
        cout<<"输出: "<<str<<endl;
    }

    Welcome they("All right!");
    they.Display();
}
```

3. Display()成员函数

要求将它设计为内联函数，也就是使用inline关键字。

```
inline void Welcome::Display()
{cout<<Get()<<endl;}
```

4. 其他成员函数

其他成员函数比较简单，设计如下：

```
void Welcome::Set(char s[])
{strcpy(str,s); }

char *Welcome::Get()
{return str;}

Welcome::~~Welcome()
{cout<<"Goodbye!"<<endl;}
```

2.4 使用指针设计类

使用指针设计类和成员函数时，要注意申请存储空间。

2.4.1 设计类

主要是设计构造函数的默认参数，即

```
Welcome(char *str="Welcome!");
```

下面是它的源程序：

```
//cpp2.h
```

```
#if !defined(CPP1_H)

#define CPP1_H

#include <iostream>
using namespace std;

class Welcome
{
private:
    char *str;
public:

    Welcome(char *str="Welcome!");
    Welcome(Welcome&);

    void Set(char *);
    void Display();
    char *Get();
    void talk();
    ~Welcome();
};

#endif
```

2.4.2 设计成员函数

1. 注意事项

申请内存时，使用strlen()函数计算长度，而且需要预留一个结束位。这样，按此方法计算出结果之后，再加1才是实际需要的长度。以构造函数为例：

```
Welcome::Welcome(char *s)
{
    str=new char[strlen(s)+1];    //增加1位
    strcpy(str,s);
}
```

在设计talk()函数时，也需要申请内存。由于不知道输入字符串的长度，所以先使用一个字符串数组接受输入，计算该字符串的长度，然后申请内存。例如：

```
void Welcome::talk()
{
    char s[100];
```