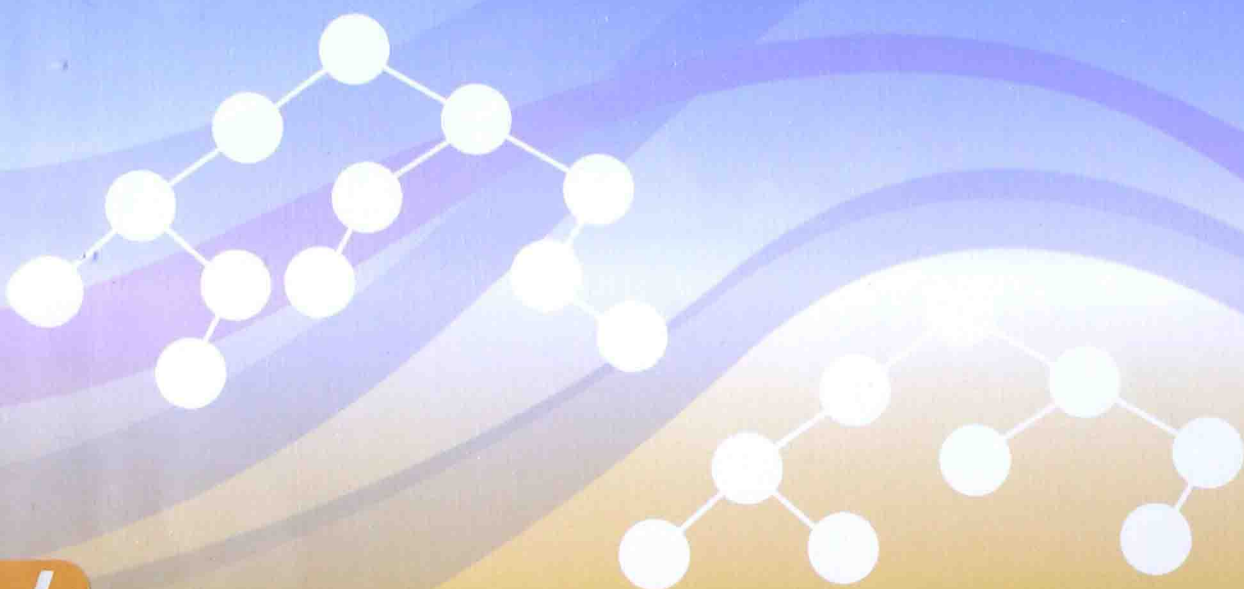




高等学校计算机教材建设立项项目

高等学校计算机专业规划教材

C++面向对象程序设计



邵兰洁 主编

马睿 徐海云 副主编



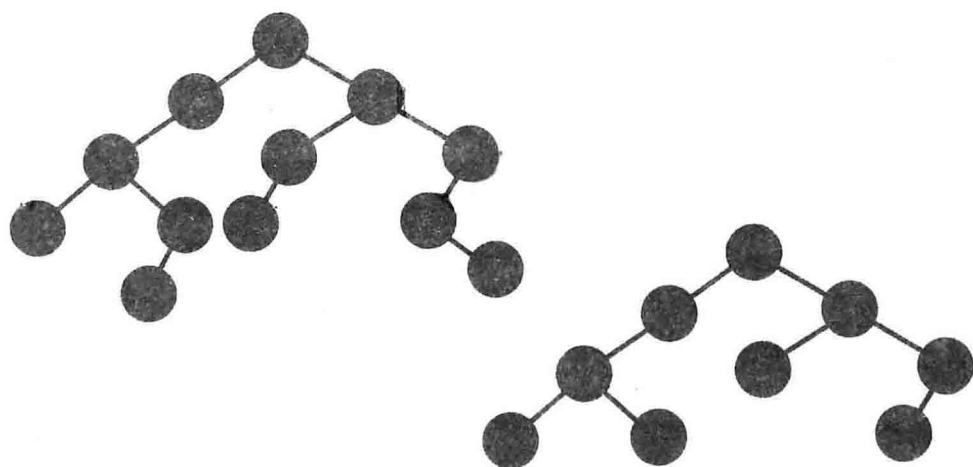
清华大学出版社

高等学校计算机专业规划教

C++面向对象程序设计

邵兰洁 主编

马睿 徐海云 副主编



清华大学出版社
北 京

内 容 简 介

本书以 CDIO 工程教育模式所倡导的“基于项目的学习”理念为指导,通过一个难度适中的综合性项目(图书馆图书借阅管理系统)的面向对象程序编制,全面而深入浅出地介绍面向对象程序设计的编程思想、标准 C++ 面向对象程序设计技术。本书主要内容包括面向对象程序设计概述,C++ 在面向过程方面对 C 语言的扩充,C++ 面向对象程序设计(类与对象、继承与派生、多态性与虚函数、友元、静态成员、运算符重载、模板与 STL、输入输出、异常处理),图形界面 C++ 程序设计等。

本书内容丰富,通俗易懂,实用性强;以项目开发为主线,将对知识点的教学融合到项目功能模块开发中,让学生“学中做”、“做中学”;学生在项目实践中学习和巩固知识点,培养和锻炼自身的自主学习、工程实践、创新、团队合作等多方面能力。

本书是按照应用型本科教学的基本要求编写的,适合作为高等院校计算机及相关专业本科生的面向对象程序设计课程教材,也适合作为具有 C 语言基础且想学习面向对象编程技术的自学者和广大程序设计人员的参考用书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

C++ 面向对象程序设计/邵兰洁主编. --北京:清华大学出版社,2015

高等学校计算机专业规划教材

ISBN 978-7-302-39253-8

I. ①C… II. ①邵… III. ①C 语言—程序设计—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2015)第 024249 号

责任编辑:龙启铭

封面设计:何凤霞

责任校对:李建庄

责任印制:杨 艳

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课件下载: <http://www.tup.com.cn>, 010-62795954

印 刷 者:北京富博印刷有限公司

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185mm×260mm 印 张:23.75 字 数:591 千字

版 次:2015 年 2 月第 1 版 印 次:2015 年 2 月第 1 次印刷

印 数:1~2000

定 价:39.00 元

产品编号:057819-01



当今,面向对象编程技术是软件开发领域的主流技术,该技术从根本上改变了人们设计软件的思维方式,它把数据和对数据的操作封装起来,集抽象性、封装性、继承性和多态性于一体,可以帮助人们开发出可靠性高、可重用性好、易修改、易扩充的软件,极大地降低了软件开发的复杂度,提高了软件开发的效率,尤其适用于功能庞大而复杂的大型软件开发。C++ 为面向对象编程技术提供全面支持,是主流的面向对象程序设计语言,在当前软件开发领域占据重要地位。全国各级各类高校计算机及相关专业基本上都开设了该课程,目的是让学生掌握面向对象程序设计的基本知识和基本技能,学会利用 C++ 语言进行面向对象程序设计,解决一般应用问题,并为后续专业课程的学习奠定程序设计基础。

C++ 由 C 语言发展而来,它在 C 语言的基础上进行功能扩充,增加了面向对象的机制。无论从编程思想、代码效率、程序的可移植性和可靠性,还是从语言基础、语言本身的实用性来讲,C++ 都是面向对象程序设计语言的典范。学好 C++, 不仅能够用于实际的程序设计,而且有助于理解面向对象程序设计的精髓,以后学习诸如 Java、C# 之类的面向对象程序设计语言也就简单了。

但是,目前的大多数 C++ 教材在内容安排上都是既介绍 C++ 的面向过程程序设计(绝大部分是介绍 C 语言的内容),又介绍 C++ 的面向对象程序设计。这样的教材对于没有 C 语言基础的读者来说是合适的。目前有不少高校是把 C 语言和 C++ 分别作为独立的两门课,尤其对计算机科学与技术专业、软件工程专业的学生来说,这样的安排更合理些。所以需要以 C 语言为起点的 C++ 教材,这样可以节省教学时间。本书就是应这种需要而产生的。本书的主要特点如下:

(1) 重点突出,内容取舍合理。本书重点讲解 C++ 的面向对象程序设计,同时兼顾 C++ 在面向过程方面对 C 语言的扩充。

(2) 通俗易懂、深入浅出。本书力求用通俗易懂的语言、生活中的现象来阐述面向对象的抽象的概念,以减少初学者学习 C++ 的困难,深入浅出,便于自学。

(3) 强调示例程序的可读性和标准化。本书的所有示例程序均遵循程序员所应该遵循的一般编程风格,如变量名、函数名和类名的命名做到“见名知义”,采用缩排格式组织程序代码并配以尽可能多的注释等,程序可读



性强。同时每个示例程序均在 Visual C++ 2012、Visual C++ 6.0、GCC 下调试通过,并给出运行结果。所有示例程序均按照标准 C++ 编写,力求培养学生从一开始就写标准 C++ 程序的习惯。

(4) 强调示例程序的实用性。本书示例程序都是经过精心设计的,实用性强,力求解决理论与实际应用脱离的矛盾,从而达到学有所用的目的。

(5) 重视学生实际编程能力的培养。本书以 CDIO 工程教育模式所倡导的“基于项目的学习”理念为指导,精心设计了一个贯穿全书各章节的综合性项目(图书馆图书借阅管理系统),兼顾教师的教与学生的学,既有利于教师示范的系统功能模块,又有利于学生练习实践的系统功能模块,非常适用于教学。随着学习进程的推进,在教师的示范与引导下,学生不断地运用所学的面向对象的 C++ 程序设计技术完成、完善该系统,最后形成一个完整的系统。学生通过项目实践练习,既理解了面向对象的编程思想,掌握了面向对象程序设计技术,又提高了自身的实践能力、自主学习能力,同时也培养了自身的创新能力、团队合作能力。

(6) 特别关注内容提醒。凡是需要学生特别关注的内容,书中都用带阴影的文本框标记,以引起学生的注意。

(7) 每章附有精彩小结。每章结束时都有一个精彩小结,对本章知识点进行高度概括,画龙点睛。

(8) 提供相应的上机指导与习题解答。相应的上机指导可以为课程上机提供方便,习题解答方便读者自查。

全书共分 11 章,第 1 章为面向对象程序设计概述,该章从一个简单的单位通信录管理系统的面向过程程序设计出发,讨论了传统的面向过程程序设计方法的不足,进而引出面向对象程序设计方法,介绍面向对象程序设计的编程思想、面向对象程序设计的基本概念、面向对象程序设计的优点、面向对象的软件开发过程。在面向对象技术理论的指导下,该章最后一节对贯穿全书的综合性项目——图书馆图书借阅管理系统进行了面向对象的分析与设计。第 2 章为 C++ 面向过程程序设计,主要介绍 C++ 在面向过程方面对 C 语言功能的扩充。第 3~10 章介绍 C++ 的面向对象程序设计,包括类与对象、继承与派生、多态性与虚函数、友元、静态成员、运算符重载、模板与 STL、输入、输出和异常处理等内容,每章最后一节均提供针对本章所讲内容的图书馆图书借阅管理系统的开发。第 11 章为图形界面 C++ 程序设计,介绍基于对话框和基于单文档图形界面 C++ 程序的设计步骤,让学生体验图形界面 C++ 程序的开发过程,消除开发窗口程序的神秘感。

本书第 1、3、4、8、11 章由邵兰洁编写,第 2 章由徐海云编写,第 5~7 章由马睿编写,第 9、10 章由张玉英编写。书中项目案例由邵兰洁设计开发。全书由邵兰洁、徐海云、马睿统稿,邵兰洁审稿。书中例题代码在不同编译环境下的测试工作由张文平完成。

本书是按照应用型本科教学的基本要求编写的,适合用作高等院校计算机及相关专业本科生的面向对象程序设计课程教材,也可用作具有 C 语言基础且想学习面向对象编程技术的自学者和广大程序设计人员的参考用书。

在本书编写过程中阅读参考了国内外大量的 C++ 书籍,这些书籍已被列在书后的参考文献中,在此谨向这些书籍的作者表示衷心的感谢。



为方便读者学习和教师教学,本书配有以下辅助资源:

- ※ 全部例题程序代码;
- ※ 配套的 PPT 电子课件;
- ※ 配套的上机指导与习题解答;
- ※ 全部习题程序代码。

以上资源可从清华大学出版社的网站(<http://www.tup.tsinghua.edu.cn>)下载或发邮件到 shaolanjie789@163.com 向作者索取。

由于编者水平有限,书中难免存在疏漏和不足之处,恳请读者批评指正。

编 者

2015 年 2 月



第 1 章 面向对象程序设计概述 /1

1.1 面向过程程序设计	1
1.2 面向对象程序设计	5
1.2.1 面向对象程序设计的思想	5
1.2.2 面向对象的基本概念	6
1.2.3 面向对象程序设计的优点	9
1.3 面向对象的软件开发	10
1.4 图书馆图书借阅管理系统的面向对象分析与设计	12
1.4.1 面向对象分析	12
1.4.2 面向对象设计	15
本章小结	16
习题	17

第 2 章 面向过程程序设计概述 /18

2.1 从 C 语言到 C++	18
2.2 简单 C++ 程序	19
2.3 C++ 对 C 语言的扩充	25
2.3.1 C++ 的输入输出	25
2.3.2 C++ 对 C 语言数据类型的扩展	26
2.3.3 常变量	26
2.3.4 指针	28
2.3.5 引用	38
2.3.6 函数	44
2.3.7 名字空间	53
2.3.8 字符串变量	56
2.3.9 复数变量	60
2.4 C++ 程序的编写和实现	63
本章小结	64
习题	64

第3章 类与对象 /66

3.1 类的声明和对象的定义	66
3.1.1 类和对象的概念及其关系	66
3.1.2 类的声明	67
3.1.3 对象的定义	68
3.2 类的成员函数	70
3.2.1 成员函数的性质	70
3.2.2 在类外定义成员函数	70
3.2.3 inline 成员函数	71
3.2.4 成员函数的存储方式	72
3.3 对象成员的访问	74
3.3.1 通过对象名和成员运算符来访问对象的成员	74
3.3.2 通过指向对象的指针来访问对象的成员	74
3.3.3 通过对象的引用来访问对象的成员	75
3.4 构造函数与析构函数	76
3.4.1 构造函数	76
3.4.2 析构函数	80
3.4.3 构造函数和析构函数的调用次序	81
3.5 对象数组	85
3.6 对象指针	88
3.6.1 指向对象的指针	88
3.6.2 指向对象成员的指针	88
3.6.3 this 指针	90
3.7 对象与 const	92
3.7.1 常对象	92
3.7.2 常对象成员	93
3.7.3 指向对象的常指针	94
3.7.4 指向常对象的指针	94
3.7.5 对象的常引用	96
3.8 对象的动态创建与释放	97
3.9 对象的赋值与复制	98
3.9.1 对象的赋值	98
3.9.2 对象的复制	102
3.9.3 对象的赋值与复制的比较	105
3.10 向函数传递对象	105
3.11 图书馆图书借阅管理系统中类的声明和对象的定义	108
本章小结	115



习题.....	115
---------	-----

第 4 章 继承与派生 /118

4.1 继承与派生的概念	118
4.2 派生类的声明	119
4.3 派生类的构成	120
4.4 派生类中基类成员的访问属性	121
4.4.1 公用继承.....	121
4.4.2 私有继承.....	123
4.4.3 保护成员和保护继承.....	125
4.4.4 成员同名问题.....	127
4.5 派生类的构造函数和析构函数	129
4.5.1 派生类构造函数.....	129
4.5.2 派生类析构函数.....	132
4.6 多重继承	134
4.6.1 声明多重继承的方法.....	134
4.6.2 多重继承派生类的构造函数与析构函数.....	134
4.6.3 多重继承引起的二义性问题.....	137
4.6.4 虚基类.....	139
4.7 基类与派生类对象的关系	143
4.8 聚合与组合	146
4.9 图书馆图书借阅管理系统中继承与聚合的应用	148
本章小结.....	165
习题.....	166

第 5 章 多态性与虚函数 /168

5.1 什么是多态性	168
5.2 向上类型转换	169
5.3 功能早绑定和晚绑定	171
5.4 实现功能晚绑定——虚函数	171
5.4.1 虚函数的定义和作用.....	172
5.4.2 虚析构函数.....	175
5.4.3 虚函数与重载函数的比较.....	177
5.5 纯虚函数和抽象类	177
5.6 图书馆图书借阅管理系统中的多态性	180
本章小结.....	187
习题.....	188

**第 6 章 友元与静态成员 /189**

6.1 封装的破坏——友元	189
6.1.1 友元函数	189
6.1.2 友元类	194
6.2 对象机制的破坏——静态成员	195
6.2.1 静态数据成员	196
6.2.2 静态成员函数	198
6.3 图书馆图书借阅管理系统中友元与静态成员的应用	201
本章小结	202
习题	202

第 7 章 运算符重载 /205

7.1 为什么要进行运算符重载	205
7.2 运算符重载的方法	207
7.3 重载运算符的规则	208
7.4 运算符重载函数作为类的成员函数和友元函数	210
7.4.1 运算符重载函数作为类的成员函数	210
7.4.2 运算符重载函数作为类的友元函数	214
7.5 几种常用运算符的重载	217
7.5.1 单目运算符“++”和“--”的重载	217
7.5.2 赋值运算符“=”的重载	221
7.5.3 流插入运算符“<<”和流提取运算符“>>”的重载	223
7.6 不同类型数据间的转换	227
7.6.1 系统预定义类型间的转换	227
7.6.2 转换构造函数	228
7.6.3 类型转换函数	231
7.7 图书馆图书借阅管理系统中的运算符重载	233
本章小结	238
习题	239

第 8 章 泛型编程 /240

8.1 函数模板	240
8.1.1 函数模板的定义	241
8.1.2 函数模板的实例化	243
8.1.3 模板参数	244
8.1.4 函数模板重载	248
8.2 类模板	251



8.2.1 类模板的定义·····	252
8.2.2 类模板的实例化·····	253
8.2.3 类模板参数·····	256
8.3 STL 简介·····	259
8.3.1 容器·····	259
8.3.2 迭代器·····	269
8.3.3 算法·····	271
8.3.4 函数对象·····	273
8.4 图书馆图书借阅管理系统中的泛型编程·····	276
本章小结·····	282
习题·····	282

第9章 输入输出 /285

9.1 C++ 的输入输出概述·····	285
9.1.1 C++ 的输入输出·····	285
9.1.2 C++ 的输入输出流·····	286
9.2 C++ 的标准输入输出流·····	288
9.2.1 C++ 的标准输出流·····	288
9.2.2 C++ 的标准输入流·····	291
9.3 输入输出运算符·····	297
9.3.1 输入运算符·····	297
9.3.2 输出运算符·····	298
9.3.3 输入与输出运算符的重载·····	298
9.4 C++ 格式输入输出·····	299
9.4.1 用流对象的成员函数控制输入输出格式·····	299
9.4.2 用控制符控制输入输出格式·····	302
9.5 文件操作与文件流·····	304
9.5.1 文件的概念·····	304
9.5.2 文件流类及文件流对象·····	304
9.5.3 文件的打开与关闭·····	305
9.5.4 对文本文件的操作·····	306
9.5.5 对二进制文件的操作·····	308
9.6 图书馆图书借阅管理系统中的文件操作·····	312
本章小结·····	313
习题·····	313

第10章 异常处理 /315

10.1 C++ 异常处理概述·····	315
----------------------	-----



10.2 C++ 异常处理的实现	316
10.3 异常与函数	322
10.3.1 在函数中处理异常	322
10.3.2 在函数调用中完成异常处理	323
10.3.3 限制函数异常	324
10.4 异常与类	324
10.4.1 构造函数、析构函数与异常处理	324
10.4.2 异常类	327
10.5 图书馆图书借阅管理系统中的异常处理	330
本章小结	332
习题	333

第 11 章 图形界面设计 /334

11.1 基于对话框的图形界面 C++ 程序设计	334
11.2 基于单文档的图形界面 C++ 程序设计	345
11.3 图书馆图书借阅管理系统的图形界面设计	364
本章小结	364
习题	365

参考文献 /366

第 1 章

面向对象程序设计概述

本章从一个简单的员工通讯录管理系统的面向过程程序设计出发,讨论了传统的面向过程程序设计方法的不足,进而引出面向对象程序设计方法,介绍面向对象程序设计的编程思想、基本概念,面向对象程序设计的优点,面向对象的软件开发过程。最后对贯穿全书的综合性项目(图书馆图书借阅管理系统)进行了面向对象的分析与设计。

1.1 面向过程程序设计

面向对象程序设计与面向过程程序设计作为两种常用的程序设计方法,有着各自的适用范围。在学习面向对象的程序设计之前,先来回顾一下面向过程的程序设计。

C 语言是一种支持面向过程程序设计的编程语言,在学习 C 语言时,所编写的每一个程序都是面向过程的。下面是一个实现某单位员工通讯录管理的面向过程的 C 程序框架。

```
/* 员工通讯录管理系统 C 语言源代码 addresslist.c */
#include<stdio.h>           /* 标准输入输出函数库 */
#include<stdlib.h>          /* 标准函数库 */
#include<string.h>          /* 字符串函数库 */
/* 宏定义 */
#define MAXSIZE 100         /* 通讯录中最多记录个数,用户可自行设置 */
/* 定义员工通讯录记录的数据结构 */
typedef struct record{      /* 标记为 record */
    char number[10];        /* 员工编号 */
    char name[11];          /* 员工姓名 */
    char department[20];    /* 所属部门 */
    char position[20];      /* 职务/岗位 */
    char office_telephone[12]; /* 座机 */
    char mobile_telephone[12]; /* 移动电话 */
    char email[30];         /* 电子邮箱 */
    char QQ[12];            /* 工作 QQ */
}Record;
/* 定义全局变量 */
Record AddressList[MAXSIZE]; /* 定义通讯录数组 */
int count=0;                 /* count 为计数器,存储当前正在处理的通讯录记录个数 */
```

```

int saveflag=0;                /* 是否需要存盘的标志变量 */
..... (其他全局变量定义略)
/* 自定义函数原型声明 */
void Init();    /* 初始化,把存放在外存中的通讯录记录信息读入 AddressList 数组中 */
void Show();    /* 浏览记录 */
void Append();  /* 追加记录 */
void Insert();  /* 插入记录 */
void Delete();  /* 删除记录 */
void Update();  /* 修改记录 */
void Search();  /* 查询记录 */
void Total();   /* 统计记录 */
void Sort();    /* 记录排序 */
void Save();    /* 保存记录 */
..... (其他自定义函数原型声明略)

int main() {
    int choice;                /* 存储用户对系统功能菜单的选择结果 */
    Init();                    /* 程序初始化 */
    system("color F0");        /* 设计程序运行窗口的背景为亮白色,前景为黑色 */
    do{
        system("cls");        /* 调用 DOS 命令,清屏,与 clrscr()功能相同 */
        /* -----显示系统功能菜单----- */
        printf("\n*****");
        printf("\n*****");
        printf("\n* *          * *");
        printf("\n* *          ###    欢迎使用员工通讯录管理系统    * *");
        printf("\n* *          * *");
        printf("\n* *          * *");
        printf("\n* *          1. Read      (浏览记录)      * *");
        printf("\n* *          2. Append    (追加记录)      * *");
        printf("\n* *          3. Insert    (插入记录)      * *");
        printf("\n* *          4. Delete    (删除记录)      * *");
        printf("\n* *          5. Update    (修改记录)      * *");
        printf("\n* *          6. Search    (查询记录)      * *");
        printf("\n* *          7. Sort      (记录排序)      * *");
        printf("\n* *          8. Total     (统计记录)      * *");
        printf("\n* *          9. Save      (保存记录)      * *");
        printf("\n* *          0. Exit      (退出系统)      * *");
        printf("\n* *          * *");
        printf("\n*****");
        printf("\n*****");
        printf("\n");
        printf("          请选择要执行的操作 (0~9):");
        scanf("%d", &choice);
        while (choice<0||choice>9)    /* 若用户选择有误 */

```

```

{   printf("选择有误,请重新输入:");
    scanf("%d",&choice);
}

switch (choice)                /* 选择功能 */
{
case 1: Show(); break;
case 2: Append(); break;
case 3: Insert(); break;
case 4: Delete(); break;
case 5: Update(); break;
case 6: Search(); break;
case 7: Sort(); break;
case 8: Total(); break;
case 9: Save();
}                                /* switch 结束 */

while( getchar() != '\n' );      /* 为了避免下次输入出错,需要清除键盘缓冲区 */
}while(choice!=0);
printf("您选择了退出系统!!!\n");
Save();                          /* 系统关闭前,保存内存的学生信息到外存文件 */
printf("谢谢使用本系统,再见!\n");
printf("按任意键退出");
getch();                          /* 让屏幕暂停 */
return 0;

}                                /* main 函数结束 */

/* 定义自定义函数 */
void init() {
    ..... (代码略)
}

..... (其他自定义函数实现代码略)

```

由上述单位员工通讯录管理系统的面向过程的 C 程序框架可以看出,面向过程的 C 程序的基本组成单位是函数,系统的每一项功能对应一个 C 函数,如果某项功能较复杂,可以对其进行功能分解,分解为多项子功能,子功能可以继续分解,直到每项子功能都足够简单,不需要再分解为止,每项子功能对应一个 C 函数,并由其上层功能所对应的 C 函数调用。main 函数负责调用顶层功能所对应的 C 函数。运用面向过程程序设计方法所设计出来的 C 程序模型如图 1-1 所示。

面向过程程序设计的基本思想为:功能分解、逐步求精、模块化、结构化。其程序结构按其功能划分为若干基本模块^①;各模块之间的关系尽可能简单,在功能上相对独立;每一个模块内部均有顺序、选择和循环 3 种基本结构组成;模块间通过调用或全局变量有机地联系起来。

^① 一个模块就是一个程序段,是能够实现某一功能,可以独立地进行编制、测试和维护的程序单位。在 C 语言中,一个模块可以对应一个自定义 C 函数。

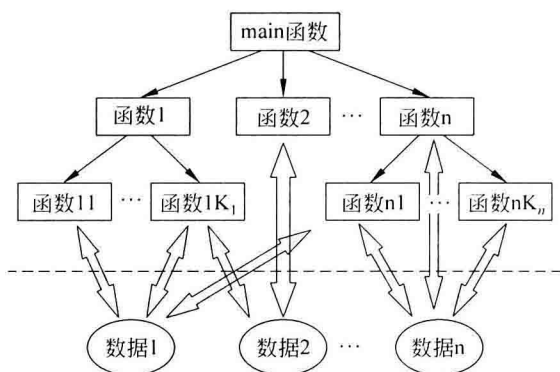


图 1-1 面向过程程序设计的程序模型

面向过程程序设计在 20 世纪 60 年代末、70 年代初从一定程度上缓解了当时的“软件危机”，它在处理较小规模的程序时比较有效。但是，随着人们对大规模软件需求的增长，面向过程的结构化程序设计逐渐显示出它的不足，具体表现如下。

1. 程序设计困难，生产率低

面向过程的程序设计是围绕功能进行的，用一个函数实现一项功能。所有数据都是公用的，一个函数可以使用任何一组数据，而一组数据又能被多个函数所使用（见图 1-1）。程序设计者必须仔细考虑每一个细节，在什么时候需要对什么数据进行操作。当程序规模较大、数据很多、操作种类繁多时，程序设计者往往感到难以应付。就如同工厂的厂长直接指挥每一个工人的工作一样，一会儿让某车间的某工人在 A 机器上用 X 材料生产轴承，一会儿又让另一车间的某工人在 B 机器上用 Y 材料生产滚珠……显然这是非常劳累的，而且往往会遗漏或搞错。所以面向过程程序设计只适用于规模较小的程序。

2. 数据不安全

在面向过程的程序中，所有数据都是公用的，谁也没有办法限制其他程序员不去修改全局数据，也不能限制其他程序员在函数中定义与全局数据同名的局部变量。因为面向过程的程序设计语言并没有提供这样一种数据保护机制。当程序规模较大时，这个问题尤其突出。

3. 程序修改困难

当要修改某个全局数据的数据结构时，所有操作该全局数据的函数都要进行修改。特别是当程序的功能因用户需求的变化而改变时，程序修改的难度更大，很有可能会导致程序的重新设计。

4. 代码重用程度低

运用面向过程程序设计方法所设计出来的程序，其基本构成单位为函数，故代码重用的粒度最大也只能到函数级。对于今天的软件开发来说，这样的重用粒度显得非常不够。

针对面向过程程序设计的不足，人们提出了面向对象程序设计方法。

1.2 面向对象程序设计

面向对象程序设计思想的出发点是思考面向过程的程序设计为什么不能有效地解决大规模程序设计的问题?实际上,面向过程的程序设计是从计算机的角度出发去解决问题,换句话说,就是用计算机的观点去观察世界。计算机的观点和人类的思维方式是有很大的区别的,从计算机的角度出发,根据对系统的分析,将系统按功能划分为一个一个的模块。在这个过程中,很多现实世界里的整体被分割成了若干个部分,这样就使得问题在从现实世界到计算机世界的转换过程中出现一定的差距。当问题较小或简单时,这种差距可以很容易地通过程序进行弥补。但当问题规模变大时,这种差距就难以弥补了。

面向对象方法的出发点是尽可能地模拟人类的思维方式去描述现实世界中的问题,使软件开发的方法尽可能接近人类认识、解决现实世界中问题的方法,使得问题在从现实世界向计算机世界转换过程中的差距尽可能的小,也就是让描述问题的问题空间和实现解法的解空间在结构上尽可能地一致。面向对象方法已经在当今的软件开发中占据了主流的位置。下面简单介绍面向对象程序设计的思想。

1.2.1 面向对象程序设计的思想

具体地讲,面向对象程序设计的基本思想如下。

(1) 客观世界中的事物都是对象(object),对象之间存在一定的关系。

面向对象方法要求从现实世界客观存在的事物出发来建立软件系统,强调直接以问题域(现实世界)中的事物为中心来思考问题和认识问题,并根据这些事物的本质特征和系统责任,把它们抽象地表示为系统中的对象,作为系统的基本构成单位。这可以使系统直接映射到问题域,保持问题域中的事物及其相互关系的本来面目。

(2) 用对象的属性(attribute)描述事物的静态特征,用对象的操作(operation)描述事物的行为(动态特征)。

(3) 对象的属性和操作结合为一体,形成一个相对独立、不可分的实体。对象对外屏蔽其内部细节,只留下少量接口,以便与外界联系。

(4) 通过抽象对对象进行分类,把具有相同属性和相同操作的对象归为一类,类是这些对象的抽象描述,每个对象是其所属类的一个实例。

(5) 复杂的对象可以用简单的对象作为其构成部分。

(6) 通过在不同程度上运用抽象的原则,可以得到一般类和特殊类。特殊类继承一般类的属性和操作,从而简化系统的构造过程。

(7) 通过关联表达类之间的静态关系。

(8) 对象之间通过传递消息进行通信,以实现对象之间的动态联系。

为了让大家对面向对象程序设计的编程思想有更深入的理解,下面先对其所涉及的基本概念进行阐述。