第十章 C++课程设计综合实践训练

10.1 课程设计简介

10.1.1 课程设计性质与教学目的

C++课程设计是计算机专业的实践必修课，是计算机本科教育的重要实践学习环节，是《C++语言程序设计》课程的继续和延伸。通过本课程设计，使学生在《C++语言程序设计》课程学习的基础上，通过完成一些具有一定难度的课程设计题目的编写、调试、运行工作，进一步掌握面向过程和面向对象程序设计的基本方法和编程技巧，加深对类与对象的理解，巩固所学理论知识，将计算机课程与实际问题相联接，使理论与实际相结合，从而能够提高学生分析问题和运用所学知识解决实际问题的能力。

一般来讲，课程设计比教学实验更复杂一些，涉及的深度更广，也更加实用。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，既能使学生巩固、加深所学理论知识，又能培养学生逻辑思维能力、动手能力和创新能力，更好地完成教学任务。另外，课程设计中较大的综合设计，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。

10.1.2 基本技能与知识背景

C++课程设计应包括 C++语言程序设计中讲过的重要的知识点及它们的综合应用。C++课程设计的核心是面向对象的编程，所以类与对象的有关设计和使用是重点，尤其要抓住抽象、封装、继承和多态等要素。要求学生的课程设计中要涉及类和继承的使用。类设计的重点是成员数据和成员函数。成员函数设计主要是其参数传递方式以及函数体的书写及功能的实现，其中需要充分利用“顺序、分支、循环”三种基本的流程控制，并需要结合数组、指针、类和结构体的使用方法来完成程序的编写。

基本的面向对象的程序设计方法需要读者掌握各种面向对象的程序设计技术，如继承和派生，虚继承、虚函数、抽象类、多态的实现、运算符重载、使用 C++标准库等。掌握在 VC++平台上进行程序设计和调试的方法。还需要掌握文件读写的相关概念和使用方法，这是程序员必备的知识，在没有学习数据库之前，编程者可以使用文件来存取数据，起到数据库的作用，所以在课程设计中须加强这方面的训练。除此之外，还要求读者初步接触软件工程的基本内容，如软件的生命周期、流程图等相关内容，为后续的学习打下基础。

10.2 课程设计教学内容

10.2.1 课程设计案例分类与操作要点

C++课程设计主要是综合 C++语言程序设计课堂上所讲过的知识点，利用面向对象的程序设计思想和方法来完成一个较为完整的项目，本章精选了三种常见的基于控制台的 C++课程设计类型：

（1）XXX 管理系统（主要完成增、删、改、查等功能）

（2）简单的模拟应用程序（如模拟自动售货机、彩票销售、股票交易等）

（3）小游戏（如贪吃蛇、俄罗斯方块等）

C++课程设计一般会在学习完整个 C++程序设计课程后集中实施。内容及学时分配如表 10-1 所示：

表 10-1 C++课程设计内容及学时分配如表

序号	设计（或实践）项目名称	内容提要	学时分配	每组人数	备注
1	案例讲解	讲解课程设计任务书中的案例并上机试做。	教学时数 3 学时，上机调试 8 学时	1	必做
2	系统分析设计	布置任务，学生选择课题并进行系统的分析与设计，包括类与界面的设计。	教学时数 3 学时，上机调试 8 学时	1~2	必做
3	系统实现及答辩	编码对系统进行实现，完成操作和接口的设计与实现，完成数据存取操作。	教学时数 2 学时，设计调试 8 学时	1~2	必做

整个课程设计的各个环节学生需要自己动手，利用学到的面向对象的基本原理和 C++语言语法以及编程技巧，通过灵活应用集成开发环境进行应用程序和系统的设计与开发，掌握面向程序设计的基本方法和步骤，强化巩固已有的编程知识，训练新的设计与编程思路，帮助熟悉程序编写，及时查究错误。写出相应的算法分析和源代码。要求上机调试通过。对课程设计进行总结，撰写课程设计报告。

在课程设计的实际操作过程中，要对系统进行正确的功能模块分析、控制模块分析；系统设计要实用；代码编写力求简练，可用，功能全面；流程图、说明书等要清楚明了；最好每个人一个题目，如果是题目比较大可以 2 个人合作完成，但一定要分清任务，文档不能抄袭。

10.2.2 报告撰写要求与格式

课程设计任务完成时上交课程设计报告和源程序。课程设计报告应包括如下几个部分内容：

1、需求分析

（1）选做此项目或课题的目的。

（2）程序所实现的功能。

2、总体设计

程序设计组成框图、流程图，根据所选题目的设计要求进行面向对象的系统分析，要求有完整系统分析过程与功能模块分析。设计思路与设计过程的阐述应详尽、明确。

3、详细设计

模块功能说明，如函数功能、入口及出口参数说明，函数调用关系描述等。

4、编写代码与测试

根据面向对象程序设计的思想和方法，编写代码并进行调试，自定义类要针对每个类成员进行注释。所用到类库中的类或系统自定义对象也要做必要的注释说明。在报告中详细书写调试方法和测试结果的分析与讨论，以及测试过程中遇到的主要问题及采取的解决措施。

5、执行结果和源程序清单

要求在报告中对执行结果进行截图，详细列出每一步的操作步骤。提供完整的程序源代码，要求每个功能模块及技术关键点、难点处加注释以说明，源代码清单可以以附录的方式放在课程设计报告的最后。

6、课程设计总结

针对整个课程设计过程进行一个总结，如系统分析过程所遇问题与解决办法，程序编写、调试运行过程与体会等等。

10.3 小型公司人员管理系统的设计与实现

10.3.1 系统描述和要求

利用C++面向对象的编程知识编写一个小型的公司人员管理系统，系统主要涉及四类人员：经理、销售经理、兼职技术人员和兼职推销员。需要存储这些人员的姓名、编号、级别、当月薪水，计算月薪总额并显示全部信息。月薪的计算方法是：经理拿固定月薪，兼职技术人员按工作小时数领取月薪，兼职推销员的报酬按该推销员当月销售额提成 ，销售经理既拿固定月薪也领取销售提成。系统要求能够按姓名或者编号增加、删除、显示、查找、和保存各类人员的信息。

10.3.2 系统分析与设计

首先确定程序至少应该具备如下功能：“查询人员”、“增加人员”、“删除人员”、“数据存盘”基本模块。人员数据可以保存到磁盘文件，那么也就意味着今后可以从磁盘文件读出人员数据，所以系统增加了“人员数据读入”模块，以方便用户使用、避免数据重复录入。考虑到系统实现简捷，人员数据文件采用文本文件，人员数据文件名：Person.txt。

Person.txt：（格式：编号,姓名,人员类别,其它数据,销售员销售额,技术人员工作小时）

1　liu　2　60000

2　wang　3　100000

3　liu　1　2000

4　wu　4　100

5　huang　2　300

6　tao　3　150000

注：人员类型编号1－经理；2－销售经理；3－销售员；4－技术人员

人员的许多固定信息如：经理、销售经理的固定月薪，销售经理、销售人员提成，技术人员小时工资等都是一些不需要每个人员都要输入的信息，所以可以将这些信息都保存在一个数据文件中，并使用“基础数据设置与修改”模块进行设置和管理。基础数据文件也采用文本文件，基础数据文件名：base.txt。系统使用的数据文件格式如下：

base.txt：

经理固定月薪　　7000

销售经理固定月薪　　5000

销售经理提成%　　1

销售人员提成%　　2

技术人员小时工资　　90

其基本格式为：项目　数据，在程序开始运行时，该文件必须存在且有初始化的内容，在程序执行中可以通过“基础数据设置与修改”选项对该文件中的数据进行修改。

在类的设计方面，系统主要涉及两个大类：公司类 Company、人员类 Person。

（1）公司类 Company：考虑系统操作的人员信息的数量具有不确定性，所以考虑使用链表保存、处理人员信息。公司类包含：所有人员信息的一个不带头结点的链表（作为数据成员）及可以对人员信息进行增，删，改，查询经营信息，基础数据设置，数据存盘，数据装入等操作的相关模块(Add,Delete,Modify,Query,Set,Save,Load)作为成员函数。

(2)人员类 Person：所有人员都具有的公共信息及操作可以使用人员类进行描述。由于系统具有 4 类人员且 4 类人员数据，操作有所不同，如销售员包含销售额，而技术人员包含工作小时数且计算工资的方法不同，所以应当为 4 类人员创建相应的 Person 类的派生类。

(3)为了使公司类可以方便处理人员信息，可以考虑将公司类确定为人员类的友元类或者人员类提供公共的方法以便公司类进行操作。为了公司类可以用共同方法操作人员类，可以将人员类的方法确定为虚函数。类的继承关系如图 10-1 所示：

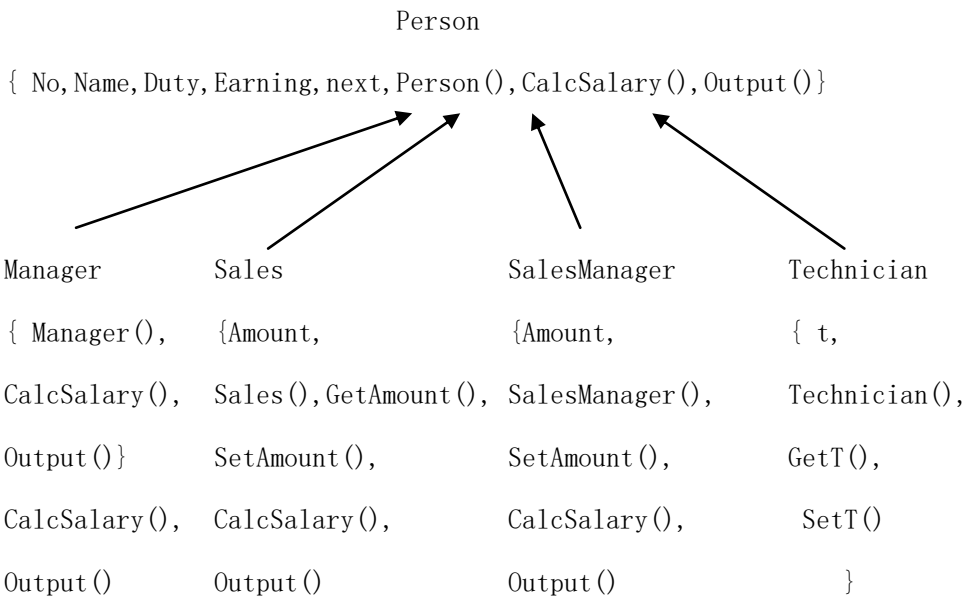


图 10-1　类的继承关系

其中：

（1）Person 类的 No-人员编号，Name-人员姓名，Duty-人员类别，Earning-工资，next-指向下一个人员的指针;Person 类的 CalcSalary()，Output()定义为纯虚函数分别表示要计算人员工资和输出人员信息,由于定义纯虚函数，所以 Person 是抽象类，具体计算工资，

输出人员信息由派生类完成。

（2）各个派生的类，包含本类对象特有的数据，Sales::Amount 为销售员销售额，SalesManager::Amount 为销售经理的总销售额（系统统计各个销售员销售额得到销售经理的总销售额），Technician::t 为技术人员工作小时数。

类 Company 中包含 person 类型的指针变量：Person *Worker，该指针变量用于人员链表，由此来完成对公司人员的增、删、改、查等一系列操作，具体函数包括 Company(),~Company(),Add(),Delete(),Modify(),Query(),Set(),Save(),Load()，Clear()，分别表示系统各个功能模块:增加人员，删除人员，修改人员，查询本月经营信息，基础数据设置与修改，数据存盘，数据读入，清除人员链表所有结点。系统关键流程图如图 10-2 到图 10-6 所示：

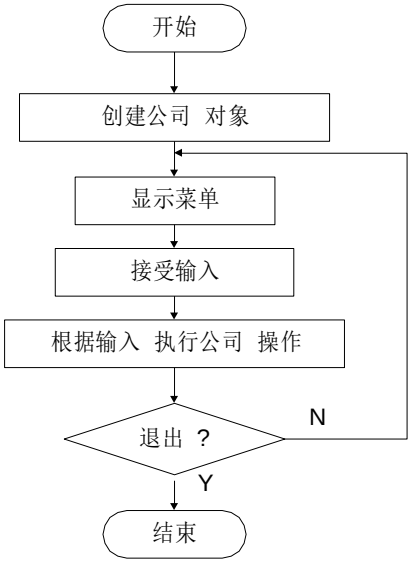


图 10-2　主函数流程

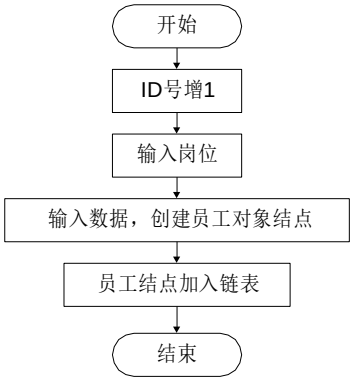


图 10-3　Company::Add()

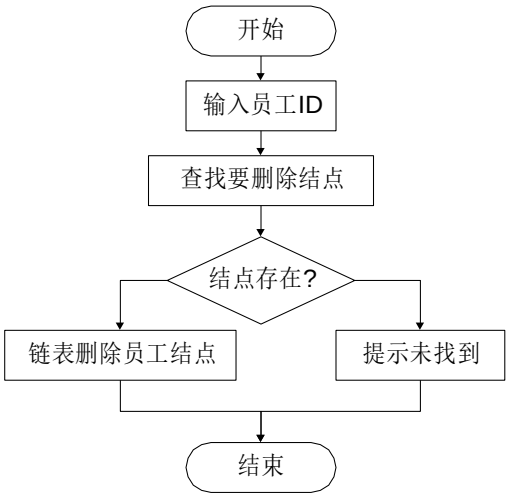


图 10-4 Company::Delete()

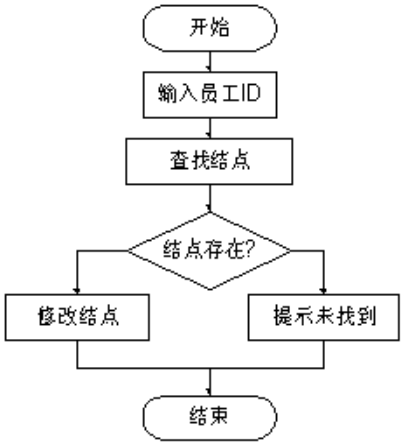


图 10-5　Company::Modify

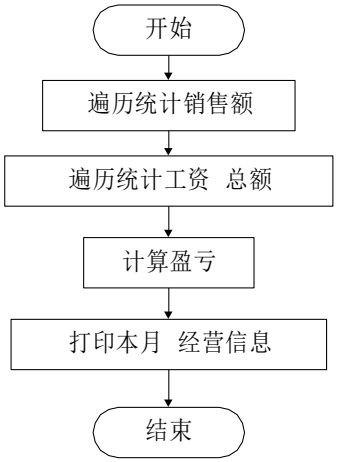


图 10-6 Company::Query()

10.3.3 系统实现

```
/*本程序有关的两个数据文件：
base.txt－基础数据文件(必须存在，且按规定格式保存)
person.txt－人员信息文件(可选)
*/

#include <iostream.h>
#include <fstream.h>
#include <ctype.h>
#include <string.h>

//全局数据，对象
double ManagerSalary;          //经理固定月薪
double SalesManagerSalary;   //销售经理固定月薪
double SalesManagerPercent; //销售经理提成%
double SalesPercent;           //销售人员提成%
double WagePerHour;            //技术人员小时工资
int ID;                        //员工标识(要保证唯一)

class Person   //员工类
{
protected:
   int No;   //编号
   char Name[20];   //姓名
   int Duty;          //岗位
   double Earning; //收入
   Person *next;
public:
   Person(char ID,char *Name,int Duty)
   {
      this->Duty=Duty;
      strcpy(this->Name,Name);
```

```
        this->No=ID;
    }
    virtual void CalcSalary()=0;
    virtual void Output()=0;
    friend class Company;
};

class Manager:public Person    //经理类
{
public:
    Manager(char ID,char *Name,int Duty):Person(ID,Name,Duty){}
    void CalcSalary(){Earning=ManagerSalary;}
    void Output()
    {
        CalcSalary();
        cout<<No<<"\t"<<Name<<"\t 经理\t"<<Earning<<endl;
    }
};

class SalesManager:public Person    //销售经理类
{
private:
    double Amount;
public:
    SalesManager(char ID,char *Name,int Duty):Person(ID,Name,Duty){}
    void SetAmount(double s)
    {
        Amount=s;
    }
    void CalcSalary()
    {
        Earning=SalesManagerSalary+Amount*SalesManagerPercent/100;
    }
    void Output()
    {
        CalcSalary();
        cout<<No<<"\t"<<Name<<"\t 销售经理\t"<<Earning<<endl;
    }
};

class Technician:public Person    //技术员类
{
private:
    double t;
```

```
public:
   Technician(char ID,char *Name,int Duty,double T):Person(ID,Name,Duty)
   {
      this->t=T;
   }
   double GetT()
   {
      return t;
   }
   void SetT(double T)
   {
      this->t=T;
   }
   void CalcSalary()
   {
      Earning=WagePerHour*t;
   }
   void Output()
   {
      CalcSalary();
      cout<<No<<"\t"<<Name<<"\t 技术员\t"<<t<<"\t"<<Earning<<endl;
   }
};

class Sales:public Person    //销售员类
{
private:
   double Amount;
public:
   Sales(char ID,char *Name,int Duty,double Amount):Person(ID,Name,Duty)
   {
      this->Amount=Amount;
   }
   double GetAmount()
   {
      return Amount;
   }
   void SetAmount(double Amount)
   {
      this->Amount=Amount;
   }
   void CalcSalary()
   {
      Earning=SalesPercent/100*Amount;
```

```
    }
    void Output()
    {
        CalcSalary();
        cout<<No<<"\t"<<Name<<"\t 销售员\t"<<Amount<<"\t"<<Earning<<endl;
    }
};

class Company    //公司类
{
private:
    Person *Worker;    //员工表
    void Clear();    //清除内存中数据
public:
    Company()
    {
        Worker=0;
        Load();
    }
    ~Company()
    {
        Person *p;
        p=Worker;
        while(p)
        {
            p=p->next;
            delete Worker;
            Worker=p;
        }
        Worker=0;
    }
    void Add();          //增加人员
    void Delete();    //删除人员
    void Modify();    //修改人员
    void Query();      //查询人员
    void Set();          //基础数据设置与修改
    void Save();       //数据存盘(包括基础数据，人员数据)
    void Load();       //数据读入(包括基础数据，人员数据)
};

void Company::Clear()    //清除内存中人员数据(内部使用)
{
    Person* p=Worker;
    while(p)
```

```
    {
        Worker=p->next;
        delete p;
        p=Worker;
    }
}

void Company::Add()
{
    Person *p;   //新结点指针
    int Duty;
    char Name[20];
    double Amount,T;

    cout<<"\n**  新增员工  **\n";

    //输入员工信息
    ID++;
    cout<<"输入岗位(1-经理 2-销售经理 3-销售员 4-技术员):";   cin>>Duty;
    cout<<"输入姓名:";   cin>>Name;
    if(Duty==3)
    {
        cout<<"本月销售额:";   cin>>Amount;
    }
    else if(Duty==4)
    {
        cout<<"本月工作小时数(0-168):";
        cin>>T;
    }

    //创建新员工结点
    switch(Duty)
    {
        case 1:p=new Manager(ID,Name,Duty); break;
        case 2:p=new SalesManager(ID,Name,Duty);   break;
        case 3:p=new Sales(ID,Name,Duty,Amount);     break;
        case 4:p=new Technician(ID,Name,Duty,T);   break;
    }
    p->next=0;

    //员工结点加入链表
    if(Worker)   //若已经存在结点
    {
        Person *p2;
```

```
        p2=Worker;
        while(p2->next)   //查找尾结点
        {
            p2=p2->next;
        }
        p2->next=p;   //连接
    }
    else   //若不存在结点(表空)
    {
        Worker=p;   //连接
    }
}

void Company::Delete()   //删除人员
{
    int No;
    cout<<"\n**  删除员工  **\n";
    cout<<"ID:";   cin>>No;

    //查找要删除的结点
    Person *p1,*p2;   p1=Worker;
    while(p1)
    {
        if(p1->No==No)
            break;
        else
        {
            p2=p1;
            p1=p1->next;
        }
    }

    //删除结点
    if(p1!=NULL)//若找到结点，则删除
    {
        if(p1==Worker)   //若要删除的结点是第一个结点
        {
            Worker=p1->next;
            delete p1;
        }
        else   //若要删除的结点是后续结点
        {
            p2->next=p1->next;
            delete p1;
```

```
        }
        cout<<"找到并删除\n";
    }
    else    //未找到结点
        cout<<"未找到!\n";
}

void Company::Modify()
{

    int No,Duty;
    char Name[20];
    double Amount,T;

    cout<<"\n**  修改员工  **\n";
    cout<<"ID:";    cin>>No;

    //查找要修改的结点
    Person *p1,*p2;    p1=Worker;
    while(p1)
    {
        if(p1->No==No)
            break;
        else
        {
            p2=p1;
            p1=p1->next;
        }
    }

    //修改结点
    if(p1!=NULL)//若找到结点
    {
        p1->Output();
        cout<<"调整岗位(1-经理 2-销售经理 3-销售员 4-技术员):";
  cin>>Duty;
        if(p1->Duty!=Duty)    //若岗位发生变动
        {
            //修改其它数据
            cout<<"输入姓名:";    cin>>Name;
            if(Duty==3)
            {
                cout<<"本月销售额:";    cin>>Amount;
            }
```

```
            else if(Duty==4)
            {
               cout<<"本月工作小时数(0-168):";
               cin>>T;
            }

            //创建新员工结点
            Person *p3;
            switch(Duty)
            {
               case 1:p3=new Manager(p1->No,Name,Duty); break;
               case 2:p3=new SalesManager(p1->No,Name,Duty);    break;
               case 3:p3=new Sales(p1->No,Name,Duty,Amount);     break;
               case 4:p3=new Technician(p1->No,Name,Duty,T);    break;
            }

            //员工结点替换到链表
            p3->next=p1->next;
            if(p1==Worker)   //若要替换的结点是第一个结点
               Worker=p3;
            else   //若要删除的结点是后续结点
               p2->next=p3;

            //删除原来的员工结点
            delete p1;
         }
         else   //若岗位没有变动
         {
            cout<<"输入姓名:";   cin>>p1->Name;
            if(Duty==3)
            {
               cout<<"本月销售额:";cin>>Amount; ((Sales *)p1)->SetAmount(Amount);
            }
            else if(Duty==4)
            {
               cout<<"本月工作小时数(0-168):";cin>>T; ((Technician *)p1)->SetT(T);
            }
         }
         cout<<"修改成功!\n";
      }
      else   //未找到结点
         cout<<"未找到!\n";
   }
```

```
void Company::Query()
{
   cout<<"\n**  查询人员本月销售信息  **\n";

   double sum=0;     //销售额总和
   Person *p=Worker;
   while(p)
   {
      if(p->Duty==3)sum+=((Sales *)p)->GetAmount();
      p=p->next;
   }

   p=Worker;
   double sum2=0;    //工资总和
   while(p)
   {
      if(p->Duty==2)((SalesManager *)p)->SetAmount(sum);
      p->Output();
      sum2+=p->Earning;
      p=p->next;
   }

   cout<<"本月盈利:"<<sum*0.20-sum2<<endl;
   cout<<"(按照 20％利润计算)\n";
}

void Company::Set()
{
   cout<<"\n**  设置基础数据  **\n";
   cout<<"经理固定月薪["<<ManagerSalary<<"元]:";
cin>>ManagerSalary;
   cout<<"销售经理固定月薪["<<SalesManagerSalary<<"元]:";
cin>>SalesManagerSalary;
   cout<<"销售经理提成["<<SalesManagerPercent<<"％]:";
cin>>SalesManagerPercent;
   cout<<"销售人员提成["<<SalesPercent<<"％]:";
cin>>SalesPercent;
   cout<<"技术人员小时工资["<<WagePerHour<<"(元/小时)]:";
cin>>WagePerHour;
   cout<<"员工标识[>="<<ID<<"]:";
cin>>ID;
}

void Company::Save()   //数据存盘(包括基础数据，人员数据),均采用文本文件
```

```
{
   ofstream fPerson,fBase;
   char c;

   cout<<"\n 保存人员和基础数据,是否继续?[Y/N]:";   cin>>c;
   if(toupper(c)!='Y')return;

   //保存人员编号、姓名、岗位
   fPerson.open("person.txt",ios::out);
   Person *p=Worker;
   while(p)
   {
      fPerson<<p->No<<"\t"<<p->Name<<"\t"<<p->Duty<<"\t";
      if(p->Duty==3)
         fPerson<<((Sales*)p)->GetAmount()<<"\t";
      else if(p->Duty==4)
         fPerson<<((Technician *)p)->GetT()<<"\t";
      fPerson<<endl;
      p=p->next;
   }
   fPerson.close();

   //保存基础数据
   fBase.open("base.txt",ios::out);
   fBase<<"经理固定月薪\t"<<ManagerSalary<<endl;
   fBase<<"销售经理固定月薪\t"<<SalesManagerSalary<<endl;
   fBase<<"销售经理提成%\t"<<SalesManagerPercent<<endl;
   fBase<<"销售人员提成%\t"<<SalesPercent<<endl;
   fBase<<"技术人员小时工资\t"<<WagePerHour<<endl;
   fBase<<"ID\t"<<ID<<endl;
   fPerson.close();

   cout<<"\n 保存人员和基础数据已经完成...\n";
}

void Company::Load()   //数据读入(包括基础数据，人员数据)
{
   //基础数据读入
   ifstream fBase;
   char buf[80];   //buf 用于保存数据文件中的注释字符串
   fBase.open("base.txt",ios::in);
   fBase>>buf>>ManagerSalary;            //经理固定月薪
   fBase>>buf>>SalesManagerSalary;   //销售经理固定月薪
   fBase>>buf>>SalesManagerPercent; //销售经理提成%
```

```
fBase>>buf>>SalesPercent;              //销售人员提成%
fBase>>buf>>WagePerHour;                //技术人员小时工资
fBase>>buf>>ID;                         //员工标识
fBase.close();

//清除内存人员数据
Clear();
//人员数据数据读入
ifstream fPerson;
Person *p=Worker;
int No;    char Name[20];    int Duty;
double Amount,T;



fPerson.open("person.txt",ios::in);
//读一条记录
fPerson>>No>>Name>>Duty;
if(Duty==3)fPerson>>Amount;
else if(Duty==4)fPerson>>T;

while(fPerson.good())
{
   //创建员工结点
   switch(Duty)
   {
      case 1:p=new Manager(No,Name,Duty);    break;
      case 2:p=new SalesManager(No,Name,Duty);    break;
      case 3:p=new Sales(No,Name,Duty,Amount);    break;
      case 4:p=new Technician(No,Name,Duty,T);    break;
   }
   p->next=0;

   //员工结点加入链表
   if(Worker)    //若已经存在结点
   {
      Person *p2;
      p2=Worker;
      while(p2->next)    //查找尾结点
      {
         p2=p2->next;
      }
      p2->next=p;    //连接
   }
   else    //若不存在结点(表空)
```

```
        {
          Worker=p;   //连接
        }

        //读下一条记录
        fPerson>>No>>Name>>Duty;
        if(Duty==3)fPerson>>Amount;
        else if(Duty==4)fPerson>>T;
      }
      fPerson.close();

      cout<<"\n 人员和基础数据已经读入...\n";
    }

    void main()
    {
      char c;
      Company a;
      do
      {
        cout<<"\n***  公司人员管理系统  ***\n";
        cout<<"1－增加人员\n";
        cout<<"2－删除人员\n";
        cout<<"3－修改人员\n";
        cout<<"4－查询本月经营信息\n";
        cout<<"5－基础数据设置\n";
        cout<<"6－数据存盘\n";
        cout<<"7－数据读入\n";
        cout<<"8－退出\t 请选择(1-8):";
        cin>>c;
        switch(c)
        {
          case '1':   a.Add();     break;
          case '2':   a.Delete();break;
          case '3':   a.Modify();break;
          case '4':   a.Query(); break;
          case '5':   a.Set();     break;
          case '6':   a.Save();   break;
          case '7':   a.Load();   break;
        }
      }while(c!='8');
    }
```

运行程序后会出现公司人员管理系统的主页面，用户可以选择其中的功能进行操作，如果选择 “1”，系统会提示用户选择要输入人员的岗位，选完后系统提示用户输入相应的姓

名，运行效果图 10-7 所示：

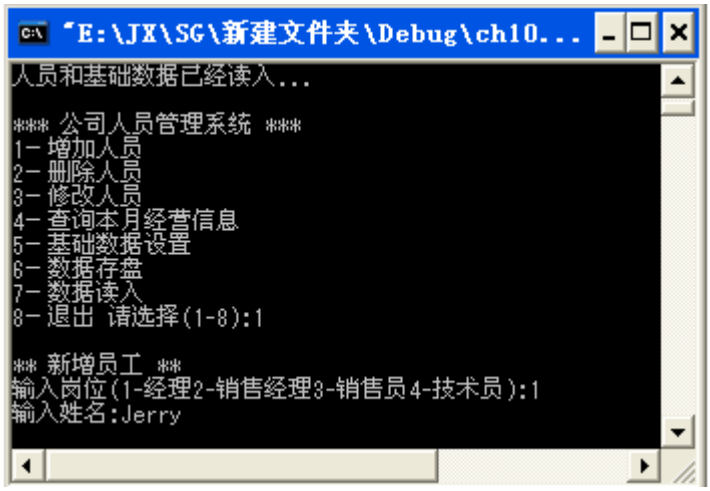


图 10-7　主界面示意图

可以多增加一些用户的信息，当增加销售人员时，系统会提示输入本月的销售额，当输入技术人员时，系统会提示用户输入技术人员工作的小时数。如图 10-8 和图 10-9 所示：

图 10-8　新增员工示意图

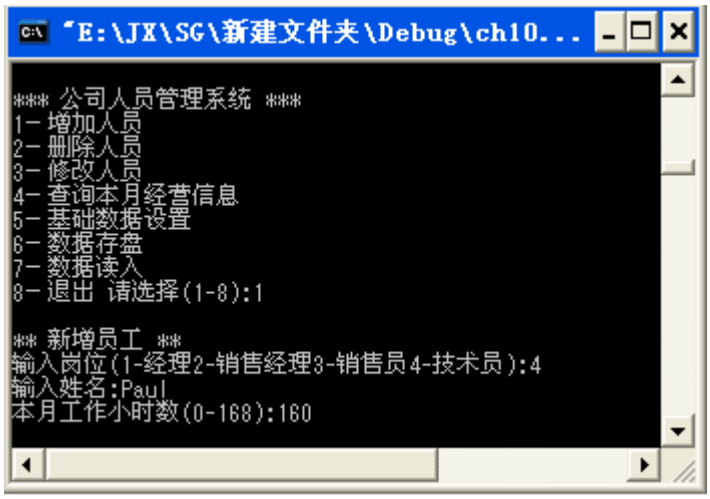


图 10-9　新增技术员示意图

这个时候打开项目文件夹中自动生成的文本文件“person”，会发现里面并没有数据，

如图 10-10 所示：

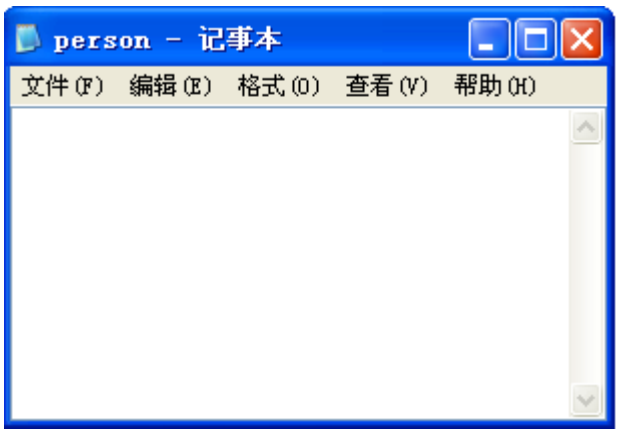


图 10-10　person.txt 文档

这时需要调用主菜单中的数据存盘功能，系统提示保存完成后才可以在文件中看到相应的数据，如图 10-11 和图 10-12 所示：

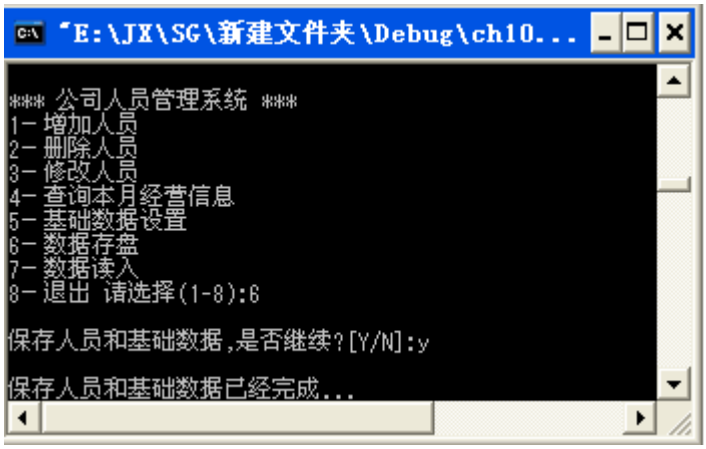


图 10-11　保存人员和基础数据示意图

图 10-12　person.txt 文档已存在数据

系统中的选项 5 为“基础数据设置”，它起到对基础数据的设置与修改的作用，在系统中输入“5”,系统会提示用户对 base.txt 文件中的数据做设置和修改，中括号中的数据是当前文件中的数据值，设置以后会将新输入的数据更新到 base.txt 文件中。具体效果如图 10-13 所示：

图 10-13　基础数据设置

系统的其余功能读者可以自行运行了解，在此不再赘述。

10.3.4 管理系统类课程设计题目

基于以上案例的讲解，读者可以在所列设计题目中任选一题于规定时间内完成设计任务。按题目要求进行系统分析与程序设计，实现题目要求的功能，程序要能正常运行，并在此基础上完成课程设计报告撰写和答辩。具体有以下几个方面的要求：

1. 查阅资料，学习新的知识和方法，培养学习能力和知识应用能力。
2. 独立思考，独立完成。培养独立思考的综合分析问题的能力。
3. 设计完成后必须提交：课程设计报告（纸质报告）、程序源代码（电子版）与编译完成的可执行文件。

题目 1 高校人员信息管理系统

1. 问题描述

某高校有四类人员：教师、实验员、行政人员、教师兼行政人员；共有的信息包括：编号、姓名、性别、年龄等。其中，教师还包含的信息有：所在系部、专业、职称；实验员还包含的信息有：所在实验室、职务；行政人员还包含的信息有：政治面貌、职称等。

2. 功能要求

1） 添加功能

程序能够任意添加上述四类人员的记录，可提供选择界面供用户选择所要添加的人员类别，要求人员的编号要唯一，如果添加了重复编号的记录时，则提示用户数据添加重复并取消添加。

2） 查询功能

可根据编号、姓名等信息对已添加的记录进行查询。如果未找到，给出相应的提示信息，如果找到，则显示相应的记录信息。

3） 显示功能

可显示当前系统中所有记录。

4） 修改功能

可根据查询结果对相应的记录进行修改，修改时注意编号的唯一性。

5） 删除功能

对已添加的人员记录进行删除。如果当前系统中没有相应的人员记录，则提示“记录为空！”并返回操作；否则输入要删除的人员的编号或姓名，根据所输入的信息删除该人员记

录，如果没有找到该人员信息，则提示相应的记录不存在。

6） 统计功能

能根据多种参数进行人员的统计。例如：统计四类人员数量以及总数，统计男、女员工的数量，统计某年龄段人员的数量等。

7） 保存功能

将当前系统中各类人员记录存入文件中。

8） 读取功能

将保存在文件中的人员信息读入到当前系统中，以供用户使用。

在完成以上基本功能的基础上，可自行进行扩展或完善。

3. 问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

1） 应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

2） 分析系统中的各个实体以及它们之间的关系；

3） 根据问题描述，设计系统的类层次；

4） 完成类层次中各个类的描述；

5） 完成类中各个成员函数的定义；

6） 完成系统的应用模块；

7） 功能调试；

8） 完成系统总结报告。

题目 2 “超市商品管理系统设计”

1、问题描述

超市中商品分为四类，分别是食品、化妆品、日用品和饮料。每种商品都包含商品名称、价格、库存量和生产厂家、品牌等信息。

主要完成对商品的销售、统计和简单管理。

2、功能要求

（1）销售功能。购买商品时，先输入类别，然后输入商品名称，并在库存中查找该商品的相关信息。如果有库存量，输入购买的数量，进行相应计算。如果库存量不够，给出提示信息，结束购买。

（2）商品简单管理功能。

添加功能：主要完成商品信息的添加。

查询功能：可按商品类别、商品名称、生产厂家进行查询。若存在相应信息，输出所查询的信息，若不存在该记录，则提示“该记录不存在！”。

修改功能：可根据查询结果对相应的记录进行修改。

删除功能：主要完成商品信息的删除。先输入商品类别，再输入要删除的商品名称，根据查询结果删除该物品的记录，如果该商品不在物品库中，则提示“该商品不存在”。

（3）统计功能。

输出当前库存中所有商品的总数及详细信息；可按商品的价格、库存量、生产厂家进行统计，输出统计信息时，要按从大到小进行排序。

（7）商品信息存盘：将当前程序中的商品信息存入文件中。

（8）读出信息：从文件中将商品信息读入程序。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 3 媒体库管理系统

1. 问题描述

图书馆中的资料很多，如果能分类对其资料流通进行管理，将会带来很多方便，因此需要有一个媒体库管理系统。

图书馆共有三大类物品资料：图书、视频光盘、图画。

这三类物品共同具有的属性有：编号、标题、作者、评级（未评级、一般、成人、儿童）等。其中图书类增加出版社、ISBN 号、页数等信息；视频光盘类增加出品人的姓名、出品年份和视频时长等信息；图画类增加出品国籍、作品的长和宽（以厘米计，整数）等信息。

2. 功能要求

1） 添加物品

程序主要完成图书馆三类物品信息的添加，要求编号唯一。如果添加了重复编号的物品时，则提示用户数据添加重复并取消添加；如果物品库已满，则提示不能再添加新的物品。

2） 查询物品

可按照三种方式进行物品的查询。

按标题查询：

按编号查询：

按类别查询：

如果未找到，给出相应的提示信息，如果找到，则显示相应的记录信息。

3） 显示物品库

可显示当前物品库中所有的物品信息。

4） 修改物品

可根据查询结果对相应的记录进行修改，修改时注意编号的唯一性。

5）删除物品

对已添加的物品信息进行删除。如果当前物品库为空，则提示“物品库为空！”并返回操作；否则输入要删除的编号，根据编号删除该物品信息，如果没有找到该物品信息，则提示“该编号不存在”。

6）统计功能

输出当前物品库中总物品数，以及按物品类别，统计出当前物品中各类别的物品数并显

示。

7） 保存物品

将当前系统中物品信息存入文件中。

8） 读取物品

将保存在文件中的物品信息读入到当前系统中，以供用户使用。

在完成以上基本功能的基础上，可自行进行扩展或完善。

3. 问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

2）分析系统中的各个实体以及它们之间的关系；

3）根据问题描述，设计系统的类层次；

4）完成类层次中各个类的描述；

5）完成类中各个成员函数的定义；

6）完成系统的应用模块；

7）功能调试；

8）完成系统总结报告。

题目 4 车辆管理系统

1. 问题描述

车辆管理系统主要负责各种车辆的常规信息管理工作。

系统中的车辆主要有大客车、小轿车和卡车。每种车辆有车辆编号、车牌号、车辆制造公司、车辆购买时间、车辆型号（大客车、小轿车和卡车）、总公里数、耗油量/公里、基本维护费用、养路费、累计总费用等信息。大客车还有载客量（最大载客数）信息，小轿车还有厢数（两厢或三厢）信息，卡车还有载重量等信息。

每台车辆当月总费用=油价*耗油量/公里+基本维护费用。

基本维护费用：客车：2000 元/月；小轿车：1000 元/月；卡车：1500 元/月

2. 功能要求

1） 添加车辆

程序主要完成车辆信息的添加，要求编号唯一。如果添加了重复编号的物品时，则提示用户数据添加重复并取消添加；如果车辆信息库已满，则提示不能再添加新的车辆信息。

2） 查询车辆

可按照三种方式进行物品的查询。

按车辆制造公司查询：

按编号查询：

按类别查询：

如果未找到，给出相应的提示信息，如果找到，则显示相应的记录信息。

3） 显示车辆信息库

可显示当前车辆信息库中所有的车辆信息。

4） 修改车辆

可根据查询结果对相应的记录进行修改，修改时注意编号的唯一性。

5）删除车辆

对已添加的车辆信息进行删除。如果当前车辆信息库为空，则提示“车辆信息库为空！”并返回操作；否则输入要删除的编号，根据编号删除该车辆信息，如果没有找到该车辆信息，则提示“该编号不存在”。

6）统计功能

输出当前车辆信息库中总车辆数，以及按车辆类别，统计出当前车辆信息库中各类别的车辆数并显示。

7） 保存车辆

将当前系统中车辆信息存入文件中。

8） 读取车辆

将保存在文件中的车辆信息读入到当前系统中，以供用户使用。

在完成以上基本功能的基础上，可自行进行扩展或完善。

3. 问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

2）分析系统中的各个实体以及它们之间的关系；

3）根据问题描述，设计系统的类层次；

4）完成类层次中各个类的描述；

5）完成类中各个成员函数的定义；

6）完成系统的应用模块；

7）功能调试；

8）完成系统总结报告。

题目 5　“学生选修课程系统设计”

1、问题描述

高校中学生信息包括：学号、姓名、性别、年龄、系别、班级、联系方式等信息。

课程信息包括：课程代码、课程名称、课程性质、总学时、学分、开课学期、选修人数等信息。学生可对课程信息进行查询，选修符合要求的课程。

根据课程信息和学生信息完成对课程的选修，需要专门的一个管理类来完成选修工作。

2、功能要求

（1）添加功能：程序能够任意添加课程和学生记录，可提供选择界面供用户选择所要添加的类别，要求编号要唯一，如果添加了重复编号的记录时，则提示数据添加重复并取消添加。

（2）查询功能：可根据编号、姓名等信息对已添加的学生和课程记录进行查询，如果未找到，给出相应的提示信息，如果找到，则显示相应的记录信息。

（3）显示功能：可显示当前系统中所有学生和课程的记录，每条记录占据一行。

（4）编辑功能：可根据查询结果对相应的记录进行修改，修改时注意编号的唯一性。

（5）删除功能：主要实现对已添加的学生和课程记录进行删除。如果当前系统中没有相应的记录，则提示“记录为空！”并返回操作。

（6）统计功能：能根据多种参数进行统计。能统计学生人数、课程的门数、选修某门

课程的学生的相关信息。

（7）保存功能：可将当前系统中各类记录存入文件中，存入方式任意。

（8）读取功能：可将保存在文件中的信息读入到当前系统中，供用户进行使用。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 6　学生成绩管理系统设计

1、问题描述

学生信息包括：学号、姓名、性别、年龄、班级等信息。

小学生除了包括学生所有信息外，还包括英语、数学和语文成绩。

中学生除了包括小学生所有信息外，还包括地理、历史成绩。

大学生除了包括学生所有信息外，还包括专业、英语、程序设计和高等数学等课程。

设计一程序能够对学生成绩进行管理，应用到继承、抽象类、虚函数、虚基类、多态和文件的输入/输出等内容。

2、功能要求

（1）添加功能：程序能够添加不同学生的记录，提供选择界面供用户选择所要添加的类别，要求学号要唯一，如果添加了重复学号的记录时，则提示数据添加重复并取消添加。

（2）查询功能：可根据学号、姓名等信息对已添加的学生记录进行查询，如果未找到，给出相应的提示信息，如果找到，则显示相应的记录信息。

（3）显示功能：可显示当前系统中所有学生的记录，每条记录占据一行。

（4）编辑功能：可根据查询结果对相应的记录进行修改，修改时注意学号的唯一性。

（5）删除功能：主要实现对已添加的学生记录进行删除。如果当前系统中没有相应的记录，则提示“记录为空！”并返回操作。

（6）统计功能：能根据多种参数进行统计。能统计学生人数、总分、单科的平均分等。

（7）保存功能：可将当前系统中各类记录存入文件中，存入方式任意。

（8）读取功能：可将保存在文件中的信息读入到当前系统中，供用户进行使用。

（9）排序功能：可按总分和单科成绩排名次。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 7　高校水电费管理系统设计

1、问题描述

住宿学生信息包括：学号、姓名、性别、年龄、班级、用电量、用水量等信息。

教工信息包括职工号、姓名、性别、年龄、工作部门、用电量、用水量等信息。

能计算出学生和教工每月所要交的电费和水费。

定义一个人员类，实现学生和教工共同的信息和行为。

2、功能要求

（1）添加功能：程序能够添加不同学生和教工的记录，提供选择界面供用户选择所要添加的类别，要求编号号要唯一，如果添加了重复编号的记录时，则提示数据添加重复并取消添加。

（2）查询功能：可根据姓名、用水量、用电量信息对已添加的学生或教工记录进行查询，如果未找到，给出相应的提示信息，如果找到，则显示相应的记录信息。

（3）显示功能：可显示当前系统中所有学生和教工的记录，每条记录占据一行。

（4）编辑功能：可根据查询结果对相应的记录进行修改，修改时注意编号的唯一性。

（5）删除功能：主要实现对已添加的学生或教工记录进行删除。如果当前系统中没有相应的记录，则提示“记录为空！”并返回操作。

（6）统计功能：能根据多种参数进行统计。能统计学生和教工的用水用电量、所要交纳的电费和水费、未交纳水电费的人员信息等。

（7）保存功能：可将当前系统中各类记录存入文件中，存入方式任意。

（8）读取功能：可将保存在文件中的信息读入到当前系统中，供用户进行使用。

（9）计算电费和水费。学生每月都有一定额度的水电是免费使用的，超过的部分需要交费。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 8　课程设计选题管理系统设计

1、问题描述

课程设计题目包括：编号、名称、关键词、实现技术、人员数（由几个人来完成）等信息。

学生信息包括：学号、姓名、性别、年龄、班级、专业等信息。

2、功能要求

（1）添加功能：程序能够添加学生的记录和课程设计题目记录，提供选择界面供用户选择所要添加的类别。添加记录时，要求学号和编号要唯一。如果添加了重复记录，则提示数据添加重复并取消添加。

（2）查询功能：可根据学号、姓名、编号、名称等信息对已添加的学生和课程设计题目进行查询，如果未找到，给出相应的提示信息，如果找到，则显示相应的记录信息。

（3）显示功能：可显示当前系统中所有学生的信息和课程设计题目信息，每条记录占据一行。

（4）编辑功能：可根据查询结果对相应的记录进行修改，修改时注意学号的唯一性。

（5）删除功能：主要实现对已添加的学生和课程设计题目记录进行删除。如果当前系统中没有相应的记录，则提示“记录为空！”并返回操作。

（6）统计功能：能根据多种参数进行统计。能按课程设计题目名称统计出学生选择该题目的人员的信息。

（7）保存功能：可将当前系统中各类记录存入文件中，存入方式任意。

（8）读取功能：可将保存在文件中的信息读入到当前系统中，供用户进行使用。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 9　公司员工考勤管理系统设计

1、问题描述

某公司需要存储雇员的编号、姓名、性别、所在部门，级别，并进行工资的计算。其中，雇员分为经理、技术人员、销售人员和销售经理。

定义一个将小时换成天数的类。转换规则：8 小时转换为一天，12 小时转换为 1.5 天。可进行天数的加、减。

定义一个记录员工生病、休假时间的类。其中包括：员工生病没工作的天数、生病可以不工作的最多天数、员工已经带薪休假的天数、员工可以带薪休假的天数。公司规定带薪休假不能超过 24 小时。生病可以不工作的最多不能超过 16 小时。

设计一程序能够对公司人员的休假情况进行管理，应用到继承、抽象类、虚函数、虚基类、多态和文件的输入/输出等内容。

2、功能要求

（1）添加功能：程序能够任意添加上述四类人员的记录，可提供选择界面供用户选择所要添加的人员类别，要求员工的编号要唯一，如果添加了重复编号的记录时，则提示数据添加重复并取消添加。还可以添加带薪休假和生病休假的记录，每条记录中必须包含员工编号和姓名。

（2）查询功能：可根据编号、姓名等信息对已添加的员工信息和休假信息进行查询，如果未找到，给出相应的提示信息，如果找到，则显示相应的记录信息；

（3）显示功能：可显示当前系统中所有记录，每条记录占据一行。

（4）编辑功能：可根据查询结果对相应的记录进行修改，修改时注意编号的唯一性。

（5）删除功能：主要实现对已添加的人员记录和休假记录进行删除。如果当前系统中没有相应的人员记录，则提示“记录为空！”并返回操作。

（6）统计功能：能根据多种参数进行人员的统计。例如，统计四类人员数量以及总数，统计任一员工的休假天数等信息。

（7）保存功能：可将当前系统中各类人员记录和休假记录存入文件中，存入方式任意。

（8）读取功能：可将保存在文件中的信息读入到当前系统中，供用户进行使用。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 10　图书管理系统设计

1、问题描述

定义图书类，属性有：书名、出版社、ISBN 号、作者、库存量、价格等信息和相关的对属性做操作的行为。

主要完成对图书的销售、统计和图书的简单管理。

2、功能要求

（1）销售功能。购买书籍时，输入相应的 ISBN 号，并在书库中查找该书的相关信息。如果有库存量，输入购买的册数，进行相应计算。如果库存量不够，给出提示信息，结束购买。

（2）图书简单管理功能。

添加功能：主要完成图书信息的添加，要求 ISBN 号唯一。当添加了重复的编号时，则提示数据添加重复并取消添加。

查询功能：可按书名、ISBN 号、作者、出版社进行查询。若存在相应信息，输出所查询的信息，若不存在该记录，则提示“该标题不存在！”。

修改功能：可根据查询结果对相应的记录进行修改，修改时注意 ISBN 号的唯一性。

删除功能：主要完成图书信息的删除。输入要删除的 ISBN 号，根据编号删除该物品的记录，如果该编号不在物品库中，则提示“该编号不存在”。

（3）统计功能。

输出当前书库中所有图书的总数及详细信息；可按书的价格、库存量、作者、出版社进行统计，输出统计信息时，要按从大到小进行排序。

（7）图书存盘：将当前程序中的图书信息存入文件中。

（8）读出信息：从文件中将图书信息读入程序。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

10.4 饮料自动售卖机模拟系统的设计与实现

10.4.1 系统描述和要求

模拟饮料自动售卖机的销售过程。顾客首先进行投币，机器显示投币金额。接下来顾客选择要购买的饮料，如果投币金额足够并且所购饮料存在，则提示用户在出口处取走饮料，同时找零。如果投币金额不足，显示提示信息。如果所购饮料已经售完，显示售完信息。

功能说明：

（1）只接受 10 元、5 元、2 元、1 元和 0.5 元的纸币和硬币。

（2）顾客一次只能投入上述一种金额的纸币或硬币，当用户重复投入时货币金额累加。

（3）销售的饮料包括 5 种：可口可乐（2 元）、百事可乐（2 元）、橙汁（3 元）、咖啡（5 元）、纯净水（1.5 元）。

（4）系统通过必要的提示信息，提示用户完成相应的操作。

（5）若顾客所购买的饮料已经售完，则进行提示并询问用户是否购买其它的饮料。

（6）完成一次售卖后，系统自动进行结算找零。

10.4.2 系统分析与设计

根据系统功能要求，首先设计处理钱币的类和商品信息类。处理钱币的类主要完成与钱币相关的工作，如给顾客找零等过程。商品信息类主要用来处理与商品相关的工作，如获得商品信息等操作。

还需要设计一个自动贩卖机类来实现饮料的售卖过程。在这个类里面，将钱币类和商品信息类作为其数据成员。同时定义了包含 5 个 GoodsInfo 对象的数组，负责保存饮料的三个信息：名称、价格和库存量，并且可以反馈这些信息。

案例需要用到类与类之间的一种关系：has-a 拥有关系，has-a 关系是指一个对象包含另一个对象，即一个对象是另一个对象的成员。

1．类的设计

根据上述的设计思想，设计了“MoneyCounter 类”、“GoodsInfo 类”和“DrinkMachine 类”3

个类。

（1） MoneyCounter 类的设计

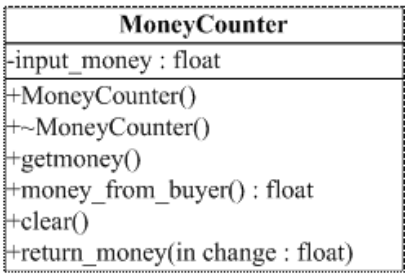


图 10-14　MoneyCounter 类图

- 数据成员

  float input_money;

  用于记录顾客投币金额
- 函数成员

  MoneyCounter();

  构造函数，初始化顾客投币金额为 0.00

  ~MoneyCounter() {}

  析构函数

  void getmoney();

  提示顾客投币

  float money_from_buyer();

  返回投币金额

  void clear();

  清空，准备下一轮投币

  void return_money(float);

  返回找的零钱

（2）GoodsInfo 类的设计

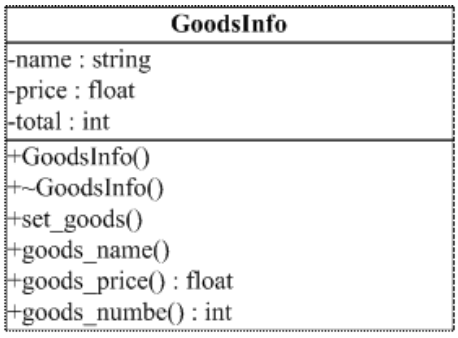


图 10-15　GoodsInfo 类图

- 数据成员

  string name;

  用于记录饮料名称

  float price;

  用于记录饮料的单价

  int　total;

  用于记录饮料的总库存数
- 函数成员

  GoodsInfo();

构造函数，初始化饮料信息
~GoodsInfo() {}
析构函数
void set_goods(string, float, int);
设置每种饮料的属性：名称，价格，数量
string goods_name();
返回饮料的名称
float goods_price();
返回饮料的价格
int goods_number();
返回饮料的数量

（3）DrinkMachine 类的设计

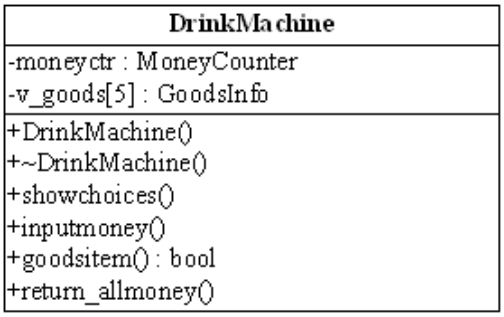


图 10-16 DrinkMachine 类图

- 数据成员

MoneyCounter moneyctr;
定义 MoneyCounter 的对象，实现投币、找零等功能
GoodsInfo v_goods[5];
定义 GoodsInfo 的对象，实现商品信息的维护，此处设计了 5 种饮料，详见该类的实现

- 函数成员

DrinkMachine();
构造函数，初始化自动售货机中的商品信息
~DrinkMachine()
析构函数
void showchoices();
显示饮料选择信息
void inputmoney();
获取顾客投入钱币
bool goodsitem(int);
检查饮料状况
void return_allmoney();
返回钱数

2．主程序设计

在主函数中，首先定义了一个 DrinkMachine 类（自动售货机类）的对象 dri，并未显式地定义 MoneyCounter 类和 GoodsInfo 类的对象。但是在 DrinkMachine 类中含有 MoneyCounter 类和 GoodsInfo 类的数据成员。

其次设计一个两重循环，外循环的持续条件是顾客继续购买，内循环的持续条件是顾客继续重复投币，即顾客可以反复投币直至投够为止。当顾客购买成功或不再继续购买时流程

中止。

10.4.3 系统实现

```
#include <iostream>
#include <string>
using namespace std;
class MoneyCounter
{
 public:
   MoneyCounter();
   ~MoneyCounter() {}
   void getmoney();                    // 显示可投币种类，提示顾客投币
   float money_from_buyer();           // 返回投币金额
   void clear();                       // 清空，准备下一轮投币
   void return_money(float);           // 返回找的零钱
 private:
   float input_money;                  // 标记顾客投币金额
 };

MoneyCounter :: MoneyCounter() : input_money(0.0f)
{                                                  // 初始化顾客投币金额为 0.00
}
void MoneyCounter :: getmoney()   //提示顾客投币
{
 float money;
 cout << "\n 请投入钱币。\n";
 cin >> money;
 input_money += money;
 cout << "\n 您投入的金额是  " << input_money <<"元。\n";
}
float MoneyCounter :: money_from_buyer() //返回顾客投币金额
{
 return input_money;
}
void MoneyCounter :: clear()
{
 input_money = 0.0f;                 //清空顾客投币金额，准备下一轮投币
}
void MoneyCounter :: return_money(float change)
{
 cout << "\n 找零  " << change << "元。\n";        //返回找零信息，
}

class GoodsInfo
{
```

```
public:
  GoodsInfo();
  ~GoodsInfo() {}
  // 设置每种饮料的属性：名称，价格，数量
  void set_goods(string, float, int);
  string goods_name();                  // 返回饮料的名称
  float goods_price();                  // 返回饮料的价格
  int goods_number();                   // 返回饮料的数量
  private:
  string name;                          // 饮料名称
  float price;                          // 饮料的单价
  int total;                            // 饮料的总库存数
};
GoodsInfo :: GoodsInfo(): name(""),price(0.0f),total(0)
{                                                   // 初始化饮料信息
}
void GoodsInfo :: set_goods(string n, float p, int num)
{                                                   // 设置饮料信息
  name = n;
  price = p;
  total = num;
}
string GoodsInfo :: goods_name() // 返回饮料名称
{
  return name;
}
float GoodsInfo :: goods_price() // 返回饮料单价
{
  return price;
}
int GoodsInfo :: goods_number()   // 返回饮料数量
{
  return total;
}

class DrinkMachine
{
public:
  DrinkMachine();                       // 初始化自动售货机中的商品信息
  ~DrinkMachine() {}
  void showchoices();                   // 显示待选饮料信息
  void inputmoney();                    // 获取顾客投入钱币
  bool goodsitem(int);                  // 检查饮料状况
  void return_allmoney();
```

```
private:
  MoneyCounter moneyctr;                // 定义 MoneyCounter 的对象
  GoodsInfo v_goods[5];          // 一共有 5 种饮料，详见该类的实现
};

DrinkMachine :: DrinkMachine()     // 初始化自动售货机中的商品信息
{
  v_goods[0].set_goods("橙汁", 3, 20);
  v_goods[1].set_goods("咖啡", 5, 0);
  v_goods[2].set_goods("纯净水", 1.5, 20);
  v_goods[3].set_goods("可口可乐", 2, 30);
  v_goods[4].set_goods("百事可乐", 2, 28);
}
void    DrinkMachine :: showchoices()          // 显示待选商品信息
{
  cout.precision(2);
  cout.setf(ios::fixed);
  cout << "\n 您投入的金额是  "<< moneyctr.money_from_buyer()<<"元。\n";
  cout << "请选择商品代码\n";
for (int i=0; i<5; i++)
{
  cout << i << "      " << v_goods[i]. goods_name()<< " " << v_goods[i].goods_price() << "元"<<
endl;
}
cout << "5     退款并且退出\n";
}
void    DrinkMachine :: inputmoney()   //显示可接受的面值，提示顾客投币
{
  cout << "\n 本机只接受 10 元、5 元、2 元、1 元和 0.5 元的纸币和硬币。";
  moneyctr.getmoney();                              // 提示顾客投币
}
// 这里 selcet，代表顾客的选择值
bool    DrinkMachine :: goodsitem(int select) // 检查货物状况
{
  int number = v_goods[select].goods_number();
  if ( number > 0 )                                  // 剩余数量>0
{
  if (moneyctr.money_from_buyer() >= v_goods[select].goods_price())       // 投币额>货物金额
{
  float change = moneyctr.money_from_buyer()-v_goods[select].goods_price();
  cout << "\n 您选择的是  " << v_goods[select].goods_name() <<"，请在出口处拿取。\n";
if ( change > 0 )                              //有找零
{
  moneyctr.return_money(change); // 显示找零信息
```

```
}
return true;
}
else
{
  cout << "\n 您投入的金额不足！ \n";
}
}
else
{
  cout << "\n 您选择的饮料已售完！ \n";
}
return false;
}
void    DrinkMachine :: return_allmoney()
{
  cout << "\n 退款  " << moneyctr.money_from_buyer() << "元。\n";
}
void main()
{
     DrinkMachine dri;
     string buf;
     bool go_on(true), cash_on(true), got_it(true);
     cout << "\n==========  欢迎使用本自动贩卖机！==========\n\n";
                //  接收投入钱币
     while ( go_on )                        //继续购买则开始下一轮循环
       {
          while ( cash_on )                  //继续投币则开始下一轮循环
          {
            dri.inputmoney();
            cout << "\n 继续投币吗？   y(yes)或者 n(no)";
            cin >> buf;
            if ( buf == "n" || buf == "no")
            {
              cash_on = false;
            }
         }
         dri.showchoices();      //  显示选择信息
         cin >> buf;        //  接收顾客的数字选择
int select = atoi(buf.c_str());
    if (select == 5)                        //显示退款信息，结束程序
       {
       dri.return_allmoney();
         go_on = false;
```

```
  }
  else
 {
 got_it = dri.goodsitem(select);
  if ( got_it )
       {
            go_on = false;          //顾客购买完毕，自动结束
        }
     else
    {
    cout << "\n 需要其他饮料吗？y(yes)或者 n(no)";
       cin >> buf;
    if ( buf == "y" || buf == "yes" )
    {
          cash_on = true;
                go_on = true;
            }
          else
         {
         dri.return_allmoney();
              go_on = false;
         }
        }
        }
    }
   cout << "\n 谢谢！再见！\n\n";
    }
```

运行系统后，系统首先要提示用户进行投币，投币结束后有是否继续投币的选项，如果输入“y”或“yes”，则提示用户继续投币，且金额可以累加，如图 10-17 所示。

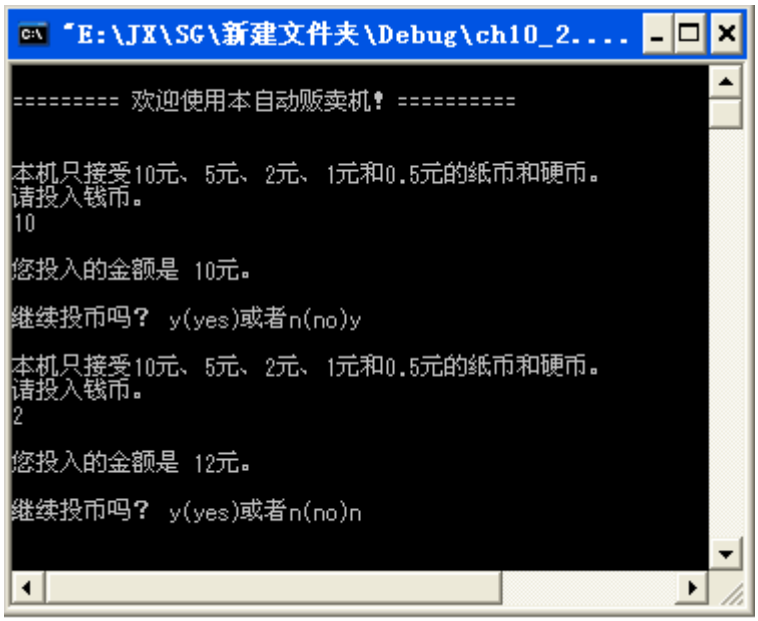
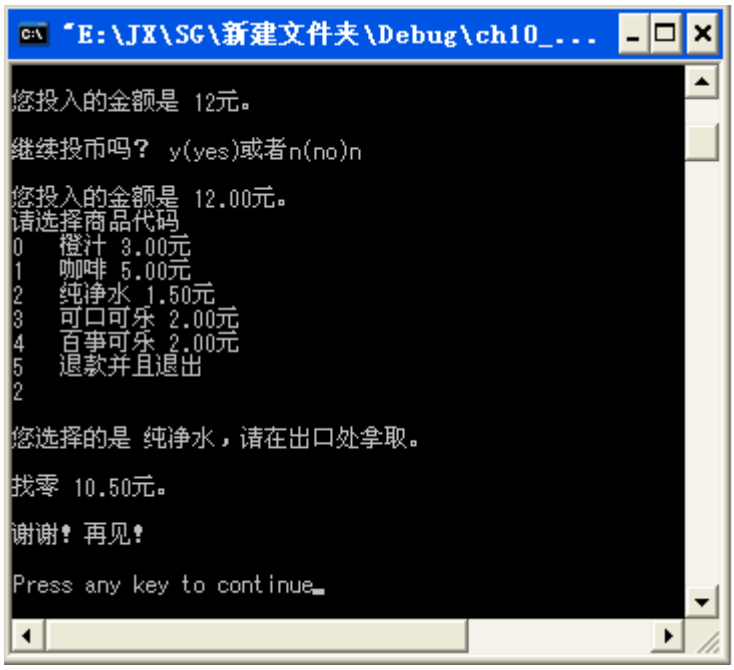


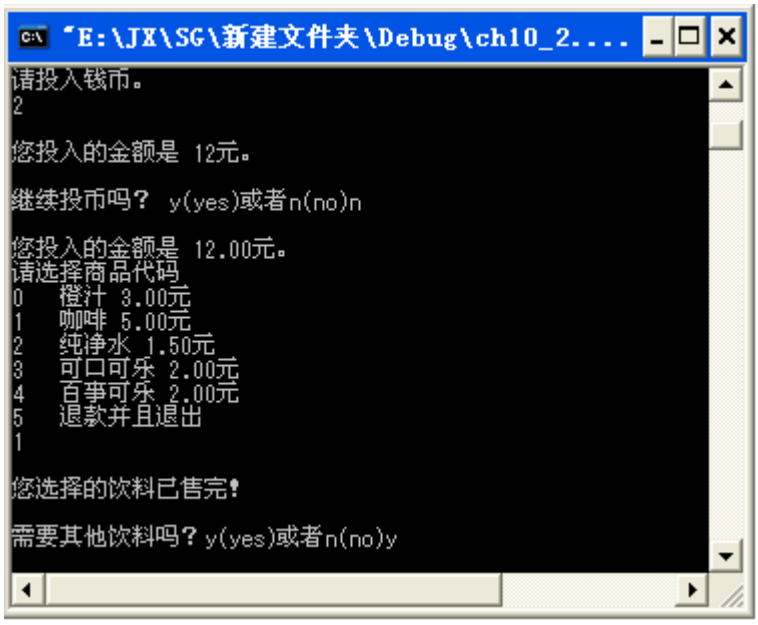
图 10-17　系统投币

当输入“n”或“no”时，结束投币，系统会提示用户投入的总金额是多少，然后出现饮料选择页面，选中相应的饮料进行购买和找零，如图 10-18 所示：



10-18 饮料购买

如果输入的饮料售卖机中已售完，系统会给出售完的提示，如图 10-19 所示：



10-19 饮料售完提示

10.4.4　系统模拟类课程设计题目

题目 1　模拟 ATM 机存取款管理系统

1、问题描述：

模拟银行的自动取款机使用过程中的界面和用户交互过程。实现查询银行卡余额、取款、修改密码、退出系统等功能。

2．功能要求：

（1）卡号、密码输入最多三次，否则直接退出系统。

（2）取款功能：取款金额受卡余额、机单笔最大取款金额及机当前剩余金额的限制。

（3）查询功能：实现查询余额功能。

（4）更改密码：实现密码更改功能，对于密码要有 2 次验证。

（5）锁卡功能：实现卡被锁的功能。

（6）退卡功能：实现退出系统功能。

3、问题的解决方案：

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 2　模拟汽车客运公司售票系统

1、问题描述：

模拟汽车客运公司售票系统中的界面和用户交互过程。实现查购票、选座、退票、改签等系统功能。

2．功能要求：

（1）客车的班次任务由调度部门确定并输入数据，一般在一段时间内不作调整。每个班次的基本信息包括：班次号，车型、发车时间，终点，座位数量、票价等。

（2）旅客购票时，应登记身份证号码、购票日期、发车日期、车次、座位号等信息。在购票时，可以查询指定发车日期、目的地的客车班次信息，在查询到的班次中，如果还有未售座位，就可以买票。旅客可以在未售座位中选择座位，也可由系统自动选择座位。购票时也可直接输入发车日期、目的地和班次，由系统自动出票，如果无票可售，则系统给与提示。座位不能重复销售，不允许售无座票。

（3）系统中应该保有从当天算起的 3 天的票源数据，开始时创建今，明，后三天的，以后每天创建后天的，每天的票源数据，应根据调度计划安排。

（4）每天的每趟班车在发售第 1 张车票时，创建这个班次的旅客登记表。

（5）售票时在旅客登记表中添加旅客信息（座位号不能重）。

（6）旅客可以办理退票，退票时即在旅客登记表中删除旅客信息。在开车前退票收取 20%退票费，开车后退票收取 50%退票费。

（7）旅客可以办理改签，在开车前可以改签同一目的地的其他车次（3 天以内），不收改签费，开车后收 20%改签费。

（8）可以输出指定班次的旅客登记表。表中包括该班次的票款合计。

3、问题的解决方案：

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；
（6）完成系统的应用模块；
（7）功能调试；
（8）完成系统总结报告。

题目 3　模拟通讯录管理系统

1、问题描述

定义通讯录类，属性有：编号、姓名、性别、通讯地址、邮箱地址、电话等信息和相关的对属性做操作的行为。

主要完成对通讯录的简单管理。

2、功能要求

（1）添加功能：程序能够添加通讯录信息，要求编号要唯一，如果添加了重复编号的记录时，则提示数据添加重复并取消添加。

（2）查询功能：可根据姓名、电话、邮箱地址等信息对已添加的信息进行查询，如果未找到，给出相应的提示信息，如果找到，则显示相应的记录信息；

（3）显示功能：可显示当前系统中所有通讯信息，每条记录占据一行。

（4）编辑功能：可根据查询结果对相应的记录进行修改，修改时注意编号的唯一性。

（5）删除功能：主要实现对已添加的通讯记录进行删除。如果当前系统中没有相应的人员记录，则提示“记录为空！”并返回操作。

（6）保存功能：可将当前系统中通讯录记录存入文件中，存入方式任意。

（7）读取功能：可将保存在文件中的信息读入到当前系统中，供用户进行使用。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；
（2）分析系统中的各个实体及它们之间的关系；
（3）根据问题描述，设计系统的类层次；
（4）完成类层次中各个类的描述；
（5）完成类中各个成员函数的定义；
（6）完成系统的应用模块；
（7）功能调试；
（8）完成系统总结报告。

题目 4 模拟分数计算器

1、问题描述

定义一个整数类。定义一个分数类，由整数类派生。能对分数进行各种计算和输入/输出。

2、功能要求

（1）定义整数类和分数类。其中，包括构造函数、析构函数、显示函数等。
（2）输入/输出：对流提取和流插入运算符进行重载。
（3）计算功能：可进行分数的加、减、乘和除法运算。
（4）化简功能：将分数化简为最简分数。
（5）异常处理功能：分数中分母不能为零。
（6）菜单功能：每种功能的操作都是在菜单中进行相应选择。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 5 模拟股票交易系统

1、问题描述

股票交易系统是一个小型的管理程序，在这个系统里，可以管理至多 5 只股票的交易。首先用户要注册，注册完后方可登陆。在登陆的界面中，管理员登陆后可以删减股票，挂起股票，解挂股票等等，通过这些功能来管理股票。同时，用户还可以查看股票情况，帮助自己进行股票的有效交易。股票的市场情况可根据用户的使用情况而随之变化。

2、功能要求

（1）首先设计主界面进行用户识别，在这里用户可以查看市场信息、注册新用户、登录用户和分析股票。然后若登录，显示下一股票操作界面，选其他有相应操作。

（2）在股票操作界面中，有买入、卖出、添加新股票、挂出股票,恢复交易、删除已有股、挂起股票,停止交易、修改代码及名称、查看等操作选择，用户可根据提示，完成相应操作。

（3）添加新股票、挂出股票,恢复交易、删除已有股、挂起股票,停止交易、修改代码及名称操作只对管理员开放。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

10.5 人机对弈游戏的设计与实现

10.5.1 系统描述和要求

利用 C++面向对象的编程知识设计一个简单的人机对弈游戏，游戏的规则是：在 3×3 的棋盘上，计算机为一方，人为一方，交替画星（‘*’）和圆圈（‘0’）。在行、列、对角线的方向上谁先连成一条直线谁就获得胜利。计算机可以动态地显示棋盘，给出提示信息和胜

负判断，并允许人选择是先下还是后下。要求实现以下功能：

（1）显示棋盘，给出提示信息和胜负结果判断标准，允许用户选择是先下还是后下。

（2）每一步都用图形化方式顺序输出。

（3）直观地表示胜负结果。

案例效果图如图 10-20 所示：

图 10-20　人机对弈效果图

10.5.2 系统分析与设计

1、系统分析

本案例的关键是要找到一种方式来表示计算机的“智能”，即计算机下棋时到底应该下哪个位置，可以用量化的思想解决该问题。

设计算机画‘*’，人画‘O’。计算机下时，应该考虑所有的空位置，并按照行、列和对角线计算每个空位的分值。在某行（列、对角线）上，按以下规则进行记分：

若已有两个连续的‘*’　　加 50 分

若已有两个连续的‘O’　　加 25 分

若已有‘*_’　　加 10 分

若已有‘O_’　　加 8 分

若已有‘_ _’　　加 4 分

（注：其中‘_’表示空格）

根据上述的记分规则，计算机每次下棋时算出每个空位置的分数，从中选择最高分的位置画‘*’。如果某条行、列、对角线上画满了 3 个‘*’则计算机赢，否则人继续下棋。

人每走一步即判断行、列、对角线上是否画满了 3 个‘O’，如画满了 3 个‘O’则人赢，否则计算机继续下棋。

如果棋盘布满了棋也未分出胜负，则和棋。

需要定义一个字符数组保存棋盘每次下完棋的状态。其中，‘*’表示计算机下的棋，‘O’表示人走的棋，数字‘1’--‘9’表示空位置。

还需要定义一个整型数组记录计算机下棋时每个空位置的分值。即如上所述的：50、25、10、8、4 分。

2、系统设计

（1）类的设计

基于上述分析，本案例定义一个类 CGame，用来处理相关的计算，打印等下棋功能。

主程序只是简单调用。

（1）CGame 类的设计

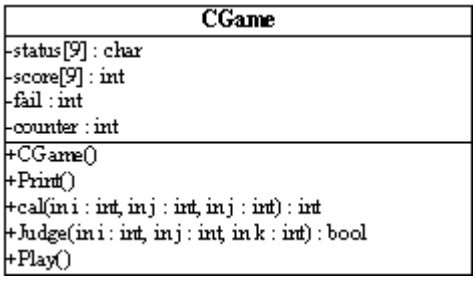


图 10-21 CGame 类图

- 数据成员

char status[9];

记录棋盘状态。

int score[9];

记录空位置的分值。

int fail;

fail 为胜败标志 0—和棋，1—人赢，2—计算机赢。

counter;

counter 为步数。

函数成员

Game();

构造函数，初始化相关变量。

Print();

打印棋盘。

int Cal(int i,int j);

计算某行的得分。

bool Judge(int i,int j,int k);

判断某条直线是否全为'O'。

Play();

启动游戏运行。

（2）主程序设计

主程序主要是调用了 CGame 类的成员函数 Play()来实现游戏的， CGame.Play()函数的流程图如图 10-22 所示。

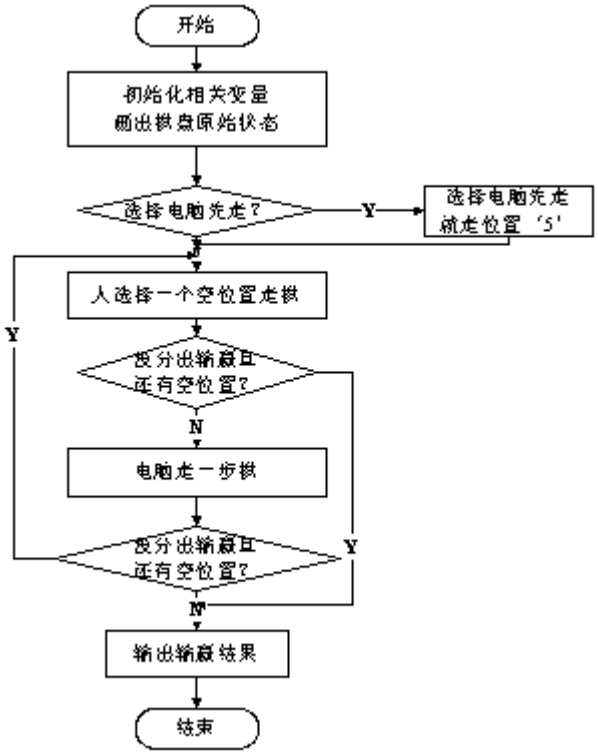


图 10-22　程序流程图

10.5.3　系统实现

```
#include <iostream.h>
class CGame          //对应游戏类
{
public:
    CGame();                                  //构造函数
    void Print();                             //打印棋盘
    int Cal(int i,int j);                     //计算某直线的得分
    bool Judge(int i,int j,int k);    //判断某直线上是否全为'O'
    Play();   //启动游戏运行
private:
    char    status[9];                        //记录棋盘状态
    int     score[9];                         //记录空位置的分值
    int     fail,counter;                     //fail 为胜败标志,counter 为步数
};
```

```
CGame::CGame()   //构造函数
 {
   for(int i=0;i<9;i++)
        status[i]='1'+i;
        counter=0;
        fail=0;
}
void CGame::Print()     //打印棋盘的格局
{
     cout<<"   "<<status[0]<<"   |   "<<status[1]<<"   |   "<<status[2]<<endl;
     cout<<"--+--+--+--+--+--"<<endl;
     cout<<"   "<<status[3]<<"   |   "<<status[4]<<"   |   "<<status[5]<<endl;
     cout<<"--+--+--+--+--+--"<<endl;
     cout<<"   "<<status[6]<<"   |   "<<status[7]<<"   |   "<<status[8]<<endl;
}

CGame::Cal(int i,int j)       //计算某直线的得分
{
   int result=0;
   if((status[i-1]=='*')&&(status[j-1]=='*'))
   {
     result=50;
     fail=2;                                                    //计算机赢
   }
   if((status[i-1]=='O')&&(status[j-1]=='O'))
              result=25;
  if(((status[i-1]=='*')&&(status[j-1]==' '))||((status[i-1]==' ')&&(status[j-1]=='*')))
            result=10;
if(((status[i-1]=='O')&&(status[j-1]==' '))||((status[i-1]==' ')&&(status[j-1]=='O')))
            result=8;
```

```
if((status[i-1]==' ')&&(status[j-1]==' '))
            result=4;
  return result;
  }
  bool CGame::Judge(int i,int j,int k)       //判断某行是否全为'O'
  {
  return ((status[i-1]=='O')&&(status[j-1]=='O')&&(status[k-1]=='O'));
}
    CGame::Play()
{
    int n=0,max=0,temp=0;;
    char ch;
    Print();     //输出棋盘原始状态
  cout<<"请选择你先下（First）还是后下(Second)(请输入 F/S)?";
  cin>>ch;
  if((ch=='S')||(ch=='s'))
  {
    status[4]='*';                    //如果计算机先走，会下到“5”的位置
    counter=counter+1;
    Print();
    cout<<"计算机下了 5!"<<endl;
  }
  do
  {
  do
     {
     cout<<"Please play(1..9)?"<<endl;
            cin>>n;
       }while(status[n-1]>='A');              //人找空位置走棋
            status[n-1]='O';
```

```
            counter++;
            Print();

if(Judge(1,2,3)||Judge(4,5,6)||Judge(7,8,9)||Judge(1,4,7)||Judge(2,5,8)||Judge(3,6,9)||Judge(1,5,9)||Judge(3,5,7))
                    fail=1;                                    //有一条直线上全为'0'，则人赢
      if((fail<1)&&(counter<9)) //计算机下'*',计算所有空位置的得分
       {
      for(int k=0;k<9;k++)
         if(status[k]!='A')            //该位置是空位置，可走棋
         {
          switch(k)
          {
     //计算各种情况下的位置得分
       case 0: score[0]=Cal(2,3)+Cal(4,7)+Cal(5,9);        //与第 1 个位置相关的直线坐标
                    break;
       case 1: score[1]=Cal(1,3)+Cal(5,8);                 //与第 2 个位置相关的直线坐标
                    break;
       case 2: score[2]=Cal(1,2)+Cal(6,9)+Cal(5,7);        //与第 3 个位置相关的直线坐标
                    break;
       case 3: score[3]=Cal(5,6)+Cal(1,7);                 //与第 4 个位置相关的直线坐标
                    break;
       case 4: score[4]=Cal(4,6)+Cal(2,8)+Cal(1,9)+Cal(3,7); //与第 5 个位置相关的直线坐标
                    break;
       case 5: score[5]=Cal(4,5)+Cal(3,9);       //与第 6 个位置相关的直线坐标
                    break;
       case 6: score[6]=Cal(8,9)+Cal(1,4)+Cal(3,5);    //与第 7 个位置相关的直线坐标
                    break;
       case 7: score[7]=Cal(7,9)+Cal(2,5);       //与第 8 个位置相关的直线坐标
                    break;
```

```
    case 8: score[8]=Cal(7,8)+Cal(3,6)+Cal(1,5);    //与第 9 个位置相关的直线坐标
                break;
        }
}
else score[k]=-1;//该位置已经使用过了
max=score[0];//选出最大分值
for(k=1;k<9;k++)
     if(score[k]>max)
     {
       max=score[k];
       temp=k;
     }
status[temp]='*';
counter++;
cout<<"计算机下了  "<<temp+1<<" !"<<endl;
Print();
}
}while((fail<=0)&&(counter<9)); //没分出输赢且还有空位置则继续下棋
      //输出人机游戏的结果
if(fail==0)
cout<<"We drew!"<<endl;
else
if(fail==1)
cout<<"你赢了!"<<endl;
else cout<<"计算机赢了!"<<endl;
}
void main()
{
   CGame game;
   game.Play();
```

}

运行系统，当计算机下棋时会根据计分规则来优先选择落子地点，从而最快的达到赢棋的目的，这在一定程度上反映了人工智能的思想，效果图如图 10-23 所示：

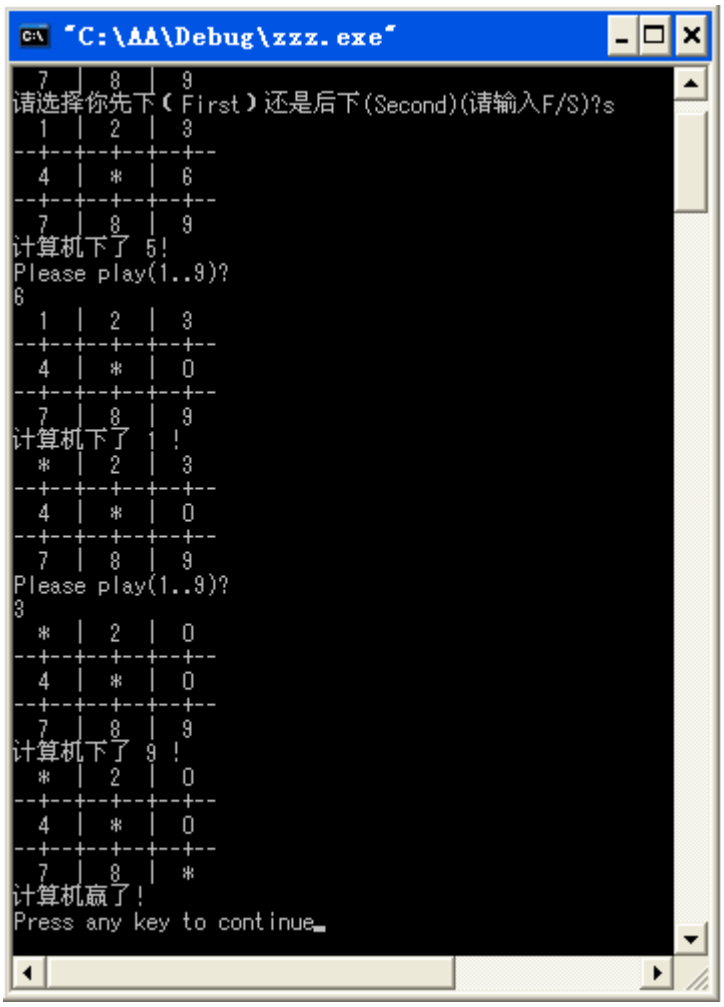


图 10-23　系统运行效果图

这只是一个简单的模拟，分数的计算是按位置来算的，随着人下棋位置的变化，计算机可能会计算出同一个位子来落子，这点等大家学了人工智能的相关知识就可以解决。

10.5.4　小游戏类课程设计题目

题目 1　五子棋游戏

1、问题描述

利用面向对象的程序思想设计一个五子棋游戏，要求玩家在游戏棋盘上逐个输入黑子或白子的坐标,谁的棋子先在行、列、对角线的方向上连成一条直线谁就获得胜利，游戏要求在 DOS 界面生成一个可供操作的棋盘。通过输入坐标完成对应落子情况，在输入过程中判断落子是否正确、是否有一方胜利等情况。

2、功能要求

（1）　输出棋盘界面菜单及图像

（2）　开始进入控制

（3） 黑白棋正确输入格式控制

（4） 判断黑白输赢控制

（5） 正确计数对羿步数及下一步所要走的棋盘界面

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 2 彩票游戏

1、问题描述

利用面向对象的程序思想设计一个彩票游戏，可以模拟体彩和福彩的投彩及开奖过程。

2、功能要求

（1）无论是开奖还是下注，福彩的 6 个号码都不能重复，请在程序中进行设置。

（2）福彩的中奖号码与其数字的顺序无关，请重新设置中奖等级。

（3）进一步完善体彩部分，体彩的中奖等级分成特等奖（数字全部吻合），一等奖（6 个连续的数字吻合）、二等奖（5 个连续的数字吻合）、三等奖（4 个连续的数字吻合）、安慰奖（2 个连续的数字吻合）。

（4）在用户类中增加资金成员，可以一次下很多注（受资金限制），每注 2 元，同时设定博彩的奖励规则，将中奖的奖金加入资金账户，具体的各个中奖等级的奖金金额自定。

（5）高级玩家可以查看计算机产生的 随机数（需要输入密码），然后据此下注，只赢不输。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 3 猜数字游戏

1、问题描述

利用面向对象的程序设计思想编程实现猜数字游戏，由电脑随机产生一个数字不重复的四位数（最高位不为零），由用户输入数字并将所猜的数与计算机自动产生的数进行比较，在每次输入数字后，显示相应的提示信息，直到玩家猜对为止。

2、功能要求

要求用 C++面向对象的知识编写程序，实现数字之间的相互比较，让用户找出计算机给出的四位数字，而用户在找出四位数字的过程中，计算机需要给出用户一些提示信息，用以帮助用户找出答案。对于猜一个各个位数不等的四位数，计算机需要在程序刚运行时，确定一个随机的四位数，且各个位数不相等。而在用户输入数字时，也需要检验用户输入的数字是否满足条件，即一个各个位数不重复的四位数。只有用户输入正确的数字后，计算机才能进行比较数字的运算，如果用户输入的数字和计算机的不相等，输出提示信息，并应从新读取用户的数字进行判断，直到用户放弃猜数字或数字猜对为止。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；

（3）根据问题描述，设计系统的类层次；

（4）完成类层次中各个类的描述；

（5）完成类中各个成员函数的定义；

（6）完成系统的应用模块；

（7）功能调试；

（8）完成系统总结报告。

题目 4　贪吃蛇游戏

1、问题描述

利用面向对象程序设计的基本思路和方法设计一个贪吃蛇的小游戏，游戏的实现完成包括游戏方面开始游戏，暂停游戏以及停止游戏，游戏帮助提示与英雄榜的显示等等。贪吃蛇的基本玩法：用一个小矩形块表示蛇的一节身体，身体每长一节，增加一个矩形块，可以用上、下、左、右键控制游戏区蛇的运动方向，使之向着食物方向运动，并吞吃食物使身体增长，每吃一次分数增加十分，每五十分增加一个等级。等级提高，“蛇”的移动速度将会增快。定义“蛇”撞“墙”或者“蛇头”碰到“蛇尾”为死亡，游戏结束。

2、功能要求

（1）贪吃蛇游戏能在 DOS 下运行。

（2）能够实现蛇身体的正常移动。

（3）能够根据按键改变蛇运动的方向。

（4）能够实现蛇吃到食物后身体增长的功能 。

（5）能够根据得分显示当前的等级，得分越高，等级越高，蛇的速度越快。

（6）游戏结束后会显示玩家的得分情况。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：

（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；

（2）分析系统中的各个实体及它们之间的关系；
（3）根据问题描述，设计系统的类层次；
（4）完成类层次中各个类的描述；
（5）完成类中各个成员函数的定义；
（6）完成系统的应用模块；
（7）功能调试；
（8）完成系统总结报告。

题目 5　俄罗斯方块游戏

1、问题描述

用面向对象的思想设计一个俄罗斯方块游戏，系统自动生成各种形状的方块，用户可以对其进行旋转变形，并通过控制方向区域的“上”、“下”、“左”、“右”来控制方块的移动，其中“上”键代表变形转换，“下”、“左”、“右”均代表方向键，而“空格”代表“快速下沉”，当方块填满一行时则自动消除，用户同时获得相应的分数，当方块到达游戏区域边框的顶端时，游戏结束。

2、功能要求

（1）利用类和文件编写。
（2）能够记录游戏得分和等级。
（3）可暂停/继续游戏，用户在不愿游戏时可以选择退出。
（4）信息提示时显示颜色变化。
（5）游戏界面包括游戏区域边框、下落方块绘制、右部计分和预览图显示。
（6）程序中应生成六种常见形状的方块。在游戏过程中可以对方块进行变形，障碍判断以及消行计分等操作。

3、问题的解决方案

根据系统功能要求，可以将问题解决分为以下步骤：
（1）应用系统分析，建立该系统的功能模块框图以及界面的组织和设计；
（2）分析系统中的各个实体及它们之间的关系；
（3）根据问题描述，设计系统的类层次；
（4）完成类层次中各个类的描述；
（5）完成类中各个成员函数的定义；
（6）完成系统的应用模块；
（7）功能调试；
（8）完成系统总结报告。