第二章：开始学习 C++

```
//ex2.1--display your name and address
#include<iostream>
int main(void)
{
	using namespace std;
	cout<<"My name is liao chunguang and I live in hunan chenzhou.\n”;
}

//ex2.2--convert the furlong units to yard uints-把浪单位换位码单位
#include<iostream>
double fur2yd(double);
int main()
{
	using namespace std;
	cout<<"enter the distance measured by furlong units:";
	double fur;
	cin>>fur;
	cout<<"convert the furlong to yard"<<endl;
	double yd;
	yd=fur2yd(fur);
	cout<<fur<<" furlong is "<<yd<<" yard"<<endl;
	return 0;

}
double fur2yd(double t)
{
	return 220*t;

}

//ex2.3-每个函数都被调用两次
#include<iostream>
void mice();
void see();
using namespace std;
int main()
{
	mice();
	mice();
	see();
	see();
	return 0;
}
```

```
void mice()
{
	cout<<"three blind mice"<<endl;
}

void see()
{
	cout<<"see how they run"<<endl;

}
```

```
//ex2.4
#include<iostream>
int main()
{
 using namespace std;
 cout<<"Enter your age:";
 int age;
 cin>>age;
 int month;
 month=age*12;
 cout<<age<<" years is "<<month<<" months"<<endl;
 return 0;
}
```

```
//ex2.5---convert the Celsius valve to Fahrenheit value
#include<iostream>
double C2F(double);
int main()
{
using namespace std;
cout<<"please enter a Celsius value:";
double C;
cin>>C;
double F;
F=C2F(C);
cout<<C<<" degrees Celsius is "<<F<<" degrees Fahrenheit."<<endl;
return 0;
}
double C2F(double t)
{
return 1.8*t+32;
}
```

```
//ex2.6---convert the light years valve to astronomical units--把光年转换为天文单位
#include<iostream>
double convert(double);//函数原型
int main()
{
using namespace std;
cout<<"Enter the number of light years:";
double light_years;
cin>>light_years;
double astro_units;
astro_units=convert(light_years);
cout<<light_years<<" light_years = "<<astro_units<<" astronomical units."<<endl;
return 0;
}
double    convert(double t)
{
return 63240*t;//1  光年=63240  天文单位
}

//ex2.7--显示用户输入的小时数和分钟数
#include<iostream>
void show();
main()
{
using namespace std;
show();
return 0;
}
void show()
{
using namespace std;
int h,m;
cout<<"enter the number of hours:";
cin>>h;
cout<<"enter the number of minutes:";
cin>>m;
cout<<"Time:"<<h<<":"<<m<<endl;


}
```

第三章：处理数据

```
//ex3.1—将身高用英尺(feet)和英寸(inch)表示
```

```
#include<iostream>
const int inch_per_feet=12;//    const 常量--1feet=12inches--1 英尺=12 英寸
int main()
{
    using namespace std;
    cout<<"please enter your height in inches:___\b\b\b";//    \b 表示为退格字符
    int ht_inch;
    cin>>ht_inch;
    int ht_feet=ht_inch/inch_per_feet;//取商
    int rm_inch=ht_inch%inch_per_feet;//取余
    cout<<"your height is "<<ht_feet<<" feet,and "
        <<rm_inch<<" inches\n";
    return 0;
}
//ex3.2--计算相应的 body mass index（体重指数）
#include<iostream>
const int inch_per_feet=12;
const double meter_per_inch=0.0254;
const double pound_per_kilogram=2.2;
int main()
{
    using namespace std;
    cout<<"Please enter your height:"<<endl;
    cout<<"First,enter your height of feet part（输入你身高的英尺部分）:_\b";
    int ht_feet;
    cin>>ht_feet;
    cout<<"Second,enter your height of inch part（输入你身高的英寸部分）:_\b";
    int ht_inch;
    cin>>ht_inch;
    cout<<"Now,please enter your weight in pound:___\b\b\b";
    double wt_pound;
    cin>>wt_pound;
    int inch;
    inch=ht_feet*inch_per_feet+ht_inch;
    double ht_meter;
    ht_meter=inch*meter_per_inch;
    double wt_kilogram;
    wt_kilogram=wt_pound/pound_per_kilogram;
    cout<<endl;
    cout<<"Your pensonal body information as follows:"<<endl;
    cout<<"身高:"<<inch<<"(英尺 inch)\n"<<"身高:"<<ht_meter<<"(米 meter)\n"
        <<"体重:"<<wt_kilogram<<"(千克 kilogram)\n";
    double BMI;
    BMI=wt_kilogram/(ht_meter*ht_meter);
```

```
        cout<<"your Body Mass Index(体重指数) is "<<BMI<<endl;
        return 0;
}

//ex3.3 以度，分，秒输入，以度输出
#include<iostream>
const int minutes_per_degree=60;
const int seconds_per_minute=60;
int main()
{
        using namespace std;
        cout<<"Enter a latitude in degrees,minutes,and seconds:\n";
        cout<<"First,enter the degrees:";
        int degree;
        cin>>degree;
        cout<<"Next,enter the minutes of arc:";
        int minute;
        cin>>minute;
        cout<<"Fianlly,enter the seconds of arc:";
        int second;
        cin>>second;
        double show_in_degree;
        show_in_degree=(double)degree+(double)minute/minutes_per_degree+(double)second/mi
nutes_per_degree/seconds_per_minute;
        cout<<degree<<"          degrees,"<<minute<<"          minutes,"<<second<<"seconds
="<<show_in_degree<<" degrees\n";
        return 0;
}

//ex3.4
#include<iostream>
const int hours_per_day=24;
const int minutes_per_hour=60;
const int seconds_per_minute=60;
int main()
{
        using namespace std;
        cout<<"Enter the number of seconds:";
        long seconds;
        cin>>seconds;
        int Day,Hour,Minute,Second;
        Day=seconds/seconds_per_minute/minutes_per_hour/hours_per_day;
        Hour=seconds/seconds_per_minute/minutes_per_hour%hours_per_day;
        Minute=seconds/seconds_per_minute%minutes_per_hour;
```

```
    Second=seconds%seconds_per_minute;
    cout<<seconds<<"seconds = "<<Day<<" days,"<<Hour<<" hours,"<<Minute<<" minutes,"<<Second<<" seconds\n";
    return 0;
}
```

```
//ex3.5
#include<iostream>
int main()
{
 using namespace std;
 cout<<"Enter the world population:";
 long long world_population;
 cin>>world_population;
 cout<<"Enter the population of the US:";
 long long US_population;
 cin>>US_population;
 double percentage;
 percentage=(double)US_population/world_population*100;
 cout<<"The population of the US is "<<percentage<<"% of the world population.\n";
 return 0;
}
```

```
//ex3.6  汽车耗油量-美国(mpg)or 欧洲风格(L/100Km)
#include<iostream>
int main()
{
 using namespace std;
 cout<<"Enter the miles of distance you have driven:";
 double m_distance;
 cin>>m_distance;
 cout<<"Enter the gallons of gasoline you have used:";
 double m_gasoline;
 cin>>m_gasoline;
 cout<<"Your car can run "<<m_distance/m_gasoline<<" miles per gallon\n";
 cout<<"Computing by European style:\n";
 cout<<"Enter the distance in kilometers:";
 double k_distance;
 cin>>k_distance;
 cout<<"Enter the petrol in liters:";
 double k_gasoline;
 cin>>k_gasoline;
 cout<<"In European style:"<<"your can used "<<100*k_gasoline/k_distance<<" liters of petrol per 100 kilometers\n";
```

```
 return 0;
}


//ex3.7    automobile gasoline consumption-耗油量--欧洲风格(L/100Km)转换成美国风格(mpg)
#include<iostream>
int main()
{
      using namespace std;
      cout<<"Enter the automobile gasoline consumption figure in\n"
            <<"European style(liters per 100 kilometers):";
      double Euro_style;
      cin>>Euro_style;
      cout<<"Converts to U.S. style(miles per gallon):"<<endl;
      cout<<Euro_style<<" L/100Km = "<<62.14*3.875/Euro_style<<" mpg\n";
      return 0;
}
// Note that 100 kilometers is 62.14 miles, and 1 gallon is 3.875 liters.
//Thus, 19 mpg is about 12.4 L/100Km, and 27 mpg is about 8.7 L/100Km.
Enter the automobile gasoline consumption figure in
European style(liters per 100 kilometers):12.4
Converts to U.S. style(miles per gallon):
12.4 L/100Km = 19.4187 mpg
Press any key to continue
// ex3.7    automobile gasoline consumption-耗油量--美国风格(mpg)转换成欧洲风格(L/100Km)
#include<iostream>
int main()
{
      using namespace std;
      cout<<"Enter the automobile gasoline consumption figure in\n"
            <<"U.S. style(miles per gallon):";
      double US_style;
      cin>>US_style;
      cout<<"Converts to European style(miles per gallon):"<<endl;
      cout<<US_style<<" mpg = "<< 62.14*3.875/US_style<<"L/100Km\n";
      return 0;
}
// Enter the automobile gasoline consumption figure in
U.S. style(miles per gallon):19
Converts to European style(miles per gallon):
19 mpg = 12.6733L/100Km
Press any key to continue
```

第四章 复合类型

//ex4.1 display the information of student

```
#include<iostream>
const int Asize=20;
using namespace std;
struct student//定义结构描述
{
	char firstname[Asize];
	char lastname[Asize];
	char grade;
	int age;
};
void display(student);//函数原型放在结构描述后
int main()
{

	cout<<"what is your first name?"<<endl;
	student lcg;//创建结构变量（结构数据对象）
	cin.getline(lcg.firstname,Asize);
	cout<<"what is your last name?"<<endl;
	cin.getline(lcg.lastname,Asize);
	cout<<"what letter grade do you deserve?"<<endl;
	cin>>lcg.grade;
	cout<<"what is your age?"<<endl;
	cin>>lcg.age;
	display(lcg);
	return 0;
}
void display(student name)
{

	cout<<"Name: "<<name.firstname<<","<<name.lastname<<endl;
	cout<<"Grade:"<<char(name.grade+1)<<endl;
	cout<<"Age:"<<name.age<<endl;
}
```

```
//ex4.2 use the string-class instead of char-array
#include<iostream>
#include<string>
int main()
{
	using namespace std;
```

```
    string name,dessert;
    cout<<"Enter your name: \n";
    getline(cin,name);

    cout<<"Enter your favorite dessert: \n";
    getline(cin,dessert);

    cout<<"I have some delicious "<<dessert;
    cout<<" for you, "<<name<<".\n";
    return 0;
}
```

//有时候会遇到需要按下两次回车键才能正确的显示结果，这是 vc++6.0 的一个 BUG，更改如下：else if (_Tr::eq((_E)_C, _D))

```
                    {_Chg = true;
                    _I.rdbuf()->sbumpc();//修改后的
                    break; }
```

ex4.3 输入其名和姓，并组合显示

```
#include<iostream>
#include<cstring>
const int Asize=20;
int main()
{
    using namespace std;
    char fname[Asize];
    char lname[Asize];
    char fullname[2*Asize+1];
    cout<<"Enter your first name:";//输入名字，存储在 fname[]数组中
    cin.getline(fname,Asize);
    cout<<"Enter your last name:";//输入姓，存储在 lname[]数组中
    cin.getline(lname,Asize);
    strncpy(fullname,lname,Asize);//把姓 lname 复制到 fullname 空数组中
    strcat(fullname,", ");//把“， ”附加到上述 fullname 尾部
    strncat(fullname,fname,Asize);//把 fname 名字附加到上述 fullname 尾部
    fullname[2*Asize]='\0';//为防止字符型数组溢出，在数组结尾添加结束符
    cout<<"Here's the information in a single string:"<<fullname<<endl;//显示组合结果
    return 0;
}
```

```
#define  _CRT_SECURE_NO_WARNINGS
#include <iostream>
#include <cstring>
const int Asize = 20;

int main()
```

```
{
    using namespace std;
    char firstname[Asize];
    char lastname[50];

    cout << "Enter your first name: ";
    cin.getline(firstname,Asize);
    cout << "Enter your last name: ";
    cin.getline(lastname,50);
    strcat(lastname,", ");
    strncat(lastname,firstname,Asize);
    cout << "Here's the information in a single string: "
         << lastname <<endl;
    return 0;
}
```

```
//ex4.4 使用 string 对象 存储、显示组合结果
#include<iostream>
#include<string>
int main()
{
    using namespace std;
    string fname,lname,attach,fullname;
    cout<<"Enter your first name:";
    getline(cin,fname);//note:将一行输入读取到 string 类对象中使用的是 getline(cin,str)
                       //它没有使用句点表示法，所以不是类方法
    cout<<"Enter your last name:";
    getline(cin,lname);
    attach=", ";
    fullname=lname+attach+fname;
    cout<<"Here's the information in a single string:"<<fullname<<endl;
    return 0;
}
```

```
//ex4.5 declare a struct and initialize it 声明结果并创建一个变量
#include<iostream>
const int Asize=20;
struct CandyBar
{
    char brand[Asize];
    double weight;
    int calory;
```

```
};
int main()
{
	using namespace std;
	CandyBar snack={"Mocha Munch",2.3,350};
	cout<<"Here's the information of snack:\n";
	cout<<"brand:"<<snack.brand<<endl;
	cout<<"weight:"<<snack.weight<<endl;
	cout<<"calory:"<<snack.calory<<endl;
	return 0;
}

//ex4.6 结构数组的声明及初始化
#include<iostream>
const int Asize=20;
struct CandyBar
{
	char brand[Asize];
	double weight;
	int calory;
};
int main()
{
	using namespace std;
	CandyBar snack[3]={
		{"Mocha Munch",2.3,350},
		{"XuFuJi",1.1,300},
		{"Alps",0.4,100}
	};
for(int i=0;i<3;i++)//利用 for 循环来显示 snack 变量的内容
	{
		cout<<snack[i].brand<<endl
			<<snack[i].weight<<endl
			<<snack[i].calory<<endl<<endl;
	}
	return 0;
}


//ex4.7 pizza 披萨饼
#include<iostream>
#include<string>
const int Size=20;
struct pizza//声明结构
```

```
{
	char company[Size];
	double diameter;
	double weight;
};
int main()
{
	using namespace std;
	pizza pie;//创建一个名为 pie 的结构变量
	cout<<"What's the    name of pizza company:";
	cin.getline(pie.company,Size);
	cout<<"What's the diameter of pizza:";
	cin>>pie.diameter;
	cout<<"What's    the weight of pizza:";
	cin>>pie.weight;
	cout<<"company:"<<pie.company<<endl;
	cout<<"diameter:"<<pie.diameter<<"inches"<<endl;
	cout<<"weight:"<<pie.weight<<"ounches"<<endl;
	return 0;
}


//ex4.8 pizza pie  披萨饼  使用 new 创建动态结构
#include<iostream>
#include<string>
const int Size=20;
struct pizza//声明结构
{
	char company[Size];
	double diameter;
	double weight;
};
int main()
{
	using namespace std;
	pizza *pie=new pizza;//使用 new 创建动态结构
	cout<<"What's the diameter of pizza:";
	cin>>pie->diameter;
	cin.get();//读取下一个字符
	cout<<"What's the    name of pizza company:";
	cin.get(pie->company,Size);
	cout<<"What's    the weight of pizza:";
	cin>>pie->weight;
	cout<<"diameter:"<<pie->diameter<<" inches"<<endl;
```

```
    cout<<"company:"<<pie->company<<endl;
    cout<<"weight:"<<pie->weight<<" ounches"<<endl;
    delete pie;//delete 释放内存
    return 0;
}

//ex.4.9  使用 new 动态分配数组—方法 1
#include<iostream>
#include<string>
using namespace std;
struct CandyBar
{
    string brand;
    double    weight;
    int calory;
};
int main()
{

    CandyBar *snack= new CandyBar[3];
    snack[0].brand="A";//单个初始化由 new 动态分配的内存
    snack[0].weight=1.1;
    snack[0].calory=200;
    snack[1].brand="B";
    snack[1].weight=2.2;
    snack[1].calory=400;
    snack[2].brand="C";
    snack[2].weight=4.4;
    snack[2].calory=500;


for(int i=0;i<3;i++)
    {
    cout << " brand: " << snack[i].brand << endl;
    cout << " weight: " << snack[i].weight << endl;
    cout << " calorie: " << snack[i].calory << endl<<endl;
    }
delete [] snack;
return 0;
}

//ex.4.10  数组—方法 1
#include <iostream>
int main()
```

```
{
	using namespace std;
	const int Size = 3;
	int success[Size];

	cout<<"Enter your success of the three times 40 meters running:\n";
	cin >> success[0]>>success[1]>>success[2];
	cout<<"success1:"<<success[0]<<endl;
	cout<<"success2:"<<success[1]<<endl;
	cout<<"success3:"<<success[2]<<endl;
	double average=(success[0]+success[1]+success[2])/3;
	cout<<"average:"<<average<<endl;
	return 0;
}

//ex.4.10 array—方法 2
#include <iostream>
#include <array>
int main()
{
	using namespace std;
	array<double,4>ad={0};
	cout<<"Enter your success of the three times 40 meters running:\n";
	cin >> ad[0]>>ad[1]>>ad[2];
	cout<<"success1:"<<ad[0]<<endl;
	cout<<"success2:"<<ad[1]<<endl;
	cout<<"success3:"<<ad[2]<<endl;
	ad[3]=(ad[0]+ad[1]+ad[2])/3;
	cout<<"average:"<<ad[3]<<endl;
	return 0;
}
```

第五章 循环和关系表达式

//ex.5.1

```
#include <iostream>

int main()
{
    using namespace std;
```

```
    cout<<"Please enter two integers: ";
    int num1,num2;
    cin>>num1>>num2;
    int sum=0;
    for(int temp=num1;temp<=num2;++temp)//or temp++
    sum+=temp;
    cout<<"The sum from "<<num1<<" to "<<num2<<" is "<<sum<<endl;
    return 0;
}
```

//ex.5.2

```
#include <iostream>
#include<array>

int main()
{
    using namespace std;
    array<long double,101>ad={0};
    ad[1]=ad[0]=1L;
    for(int i=2;i<101;i++)
        ad[i]=i*ad[i-1];
    for(int i=0;i<101;i++)
        cout<<i<<"! = "<<ad[i]<<endl;
    return 0;
}
```

```
#include <iostream>
#include <array>
using namespace std;

int main()
{
    array<long double, 101> multiply;
    multiply[0] = multiply[1] = 1LL;
    for (int i = 2; i <= 100; i++)
        multiply[i] = multiply[i-1]*i;
    cout << multiply[100];
    return 0;
}
```

//ex.5.3

```
#include <iostream>
int main()
```

```
{
    using namespace std;
    cout<<"Please enter an integer: ";
    int sum=0,num;
    while((cin>>num)&&num!=0)
    {
        sum+=num;
        cout<<"So far, the sum is "<<sum<<endl;
        cout<<"Please enter an integer: ";
    }
    return 0;
}
```

```
//ex.5.4
#include <iostream>
int main()
{
    using namespace std;
    double sum1,sum2;
    sum1=sum2=0.0;
    int year=0;
    while(sum2<=sum1)
    {
        ++year;
        sum1+=10;
        sum2=(100+sum2)*0.05+sum2;
    }
    cout<<"经过"<<year<<"年后，Cleo 的投资价值才能超过 Daphne 的投资价值。"<<endl;
    cout<<"此时，Cleo 的投资价值为"<<sum1<<"，而 Daphne 的投资价值为"<<sum2<<endl;
    return 0;
}
```

```
#include <iostream>
using namespace std;
```

```
int main()
{
    double Daphne = 100.0;
    double Cleo = 100.0;
    int year = 0;
    while (Cleo <= Daphne)
    {
        Daphne += 10;
        Cleo *= 1.05;
        year++;
    }
    cout << year << endl;
    return 0;
}
```

//ex.5.5

```
#include <iostream>
const int MONTHS = 12;
const                                                                    char*
months[MONTHS]={"January","February","March","April","May","June","July","August","Sept
ember","October","November","December"};
int main()
{
    using namespace std;
    int sales[MONTHS],sum=0;
    for(int i=0;i<MONTHS;i++)
    {
       cout<<"请输入在"<<months[i]<<"的C++ For Fools的销售量：";
       cin>>sales[i];
       sum+=sales[i];
    }
    cout<<"这一年中的C++ For Fools的总销售量为："<<sum<<endl;
    return 0;
}
```

//ex.5.6

```
#include <iostream>
const int MONTHS = 12;
const                                                                    char*
```

```
months[MONTHS]={"January","February","March","April","May","June","July","August","Sept
ember","October","November","December"};
const char* years[3]={"第一年","第二年","第三年"};
int main()
{
    using namespace std;
    int year_sale[3],sum=0,sales[3][MONTHS];
    for(int i=0;i<3;i++)
    {
            int temp=0;
            cout<<years[i]<<"的每个月销售量:"<<endl;
            for(int j=0;j<MONTHS;j++)
            {
                    cout<<"请输入"<<months[j]<<"的销售量:";
                    cin>>sales[i][j];
                    temp+=sales[i][j];
            }
            year_sale[i]=temp;
            sum+=year_sale[i];
    }
    for(int i=0;i<3;i++)
    cout<<years[i]<<"的销售量为："<<year_sale[i]<<endl;
    cout<<"这三年的总销售量为："<<sum<<endl;
    return 0;
}
 #include <iostream>
using namespace std;
const int Years = 3, Months = 12;
const char* months[Months] =
{"January","February","March","April","May","June",
    "July","August","September","October","November","December"};
int main()
{
    int sale[Years][Months] = {0};
    for (int i = 0; i < Years; i++)
    {
        int sum = 0;
        for (int j = 0; j < Months; j++)
        {
            cout << "Enter the salesment of " << months[j] << ": ";
            cin >> sale[i][j];
            sum += sale[i][j];
        }
        cout << "Salesment for this year: " << sum << endl << endl;
```

```
    }
    return 0;
}
```

//ex.5.7

```
#include <iostream>
#include <string>
using namespace std;
struct car{
       string name;
       int year;
};
int main()
{
    cout<<"How many cars do you wish to catalog? ";
    int num;
    (cin>>num).get();
    car* ps=new car[num];
    for(int i=0;i<num;++i)
    {
            cout<<"Car #"<<i+1<<":\n";
            cout<<"Please enter the make: ";
            getline(cin,ps[i].name);
            cout<<"Please enter the year made: ";
            (cin>>ps[i].year).get();
    }
    cout<<"Here is your collection:\n";
    for(int i=0;i<num;++i)
    cout<<ps[i].year<<" "<<ps[i].name<<endl;
    delete [] ps;
    return 0;
}
```

```
#include <iostream>
#include <string>
using namespace std;
struct car
{
    string maker;
    int year;
};
int main()
{
```

```
    int number;
    cout << "How many cars do you wish to catalog? ";
    cin >> number;
    car* a = new car[number];
    for (int i = 0; i < number; i++)
    {
        cout << "Car #" << i+1 << ": " << endl;
        cout << "Please enter the maker: ";
        cin.get();
        getline(cin,a[i].maker);
        cout << "Please enter the year made: ";
        cin >> a[i].year;
    }
    cout << "Here is your collection: " << endl;
    for (int i = 0; i < number; i++)
    cout << a[i].year << " " << a[i].maker <<endl;
    delete [] a;
    return 0;

}
```

```
#include <iostream>
using namespace std;
struct car
{
    char maker[20];
    int year;
};
int main()
{
    int number;
    cout << "How many cars do you wish to catalog? ";
    cin >> number;
    car* a = new car[number];
    for (int i = 0; i < number; i++)
    {
        cout << "Car #" << i+1 << ": " << endl;
        cout << "Please enter the maker: ";
        cin.get();
        cin.getline(a[i].maker, 20);
        cout << "Please enter the year made: ";
        cin >> a[i].year;
    }
    cout << "Here is your collection: " << endl;
    for (int i = 0; i < number; i++)
```

```
    cout << a[i].year << " " << a[i].maker <<endl;
    delete [] a;
    return 0;
}
```

//ex.5.8

```
#include <iostream>
#include <cstring>

int main()
{
    using namespace std;

    char word[20];
    int sum=0;
    cout<<"Enter words (to stop,type the word done):\n";
    cin>>word;
    while(strcmp(word,"done"))
    {
           sum++;
           cin>>word;
    }
    cout<<"You entered a total of "<<sum<<" words.\n";
    return 0;
}
```

//ex.5.9

```
#include <iostream>
#include <string>

int main()
{
    using namespace std;
    string word;
    int sum=0;
    cout<<"Enter words (to stop, type the word done):\n";
    cin>>word;
    while(word!="done")
    {
         sum++;
         cin>>word;
    }
    cout<<"You entered a total of "<<sum<<" words.\n";
```

```
    return 0;
}
```

和 ex.5.8 的区别是：word != "done"，因为当 word = done 一样时，返回值为 1，不一样时才是返回 0.

//ex.5.10

```
#include <iostream>
int main()
{
    using namespace std;
    cout<<"Enter number of rows:";
     int num;
     cin>>num;
     for(int i=0;i<num;i++)
     {
          for(int j=num-i;j>1;j--)
         cout<<".";
          for(int k=0;k<=i;++k)
              cout<<"*";
               cout<<endl;
     }
     return 0;
}
```

第六章 分支语句和逻辑运算符

//ex.6.1

```
#include <iostream>
#include <cctype>

int main()
{
    using namespace std;

    char ch;
    cin.get(ch);

    while(ch!='@')
    {
            if(isdigit(ch))
            cin.get(ch);
```

```
            else
            {
                if(islower(ch))
                ch=toupper(ch);
                else
                    ch=tolower(ch);
                    cout<<ch;
                    cin.get(ch);
             }
    }
    return 0;
}
```

```
#include <iostream>
#include <cctype>
using namespace std;

int main()
{
    char ch;
    cout << "Please enter: \n";
    while (cin.get(ch) && ch != '@')
    {
        if(islower(ch))
        {
            ch = toupper(ch);
            cout << ch;
        }
        else if(isupper(ch))
        {
            ch = tolower(ch);
            cout << ch;
        }
        else
            cout << ch;
    }
    return 0;
}
```

//ex.6.2--数组

```
#include <iostream>
#include<cctype>
int main()
{
```

```
    using namespace std;

    double sum=0,average=0;
    double num[10];
    int i=0,total=0;

    double temp;

    while(cin>>temp&&i<10&&!isdigit(temp))
    {
         num[i]=temp;
         sum+=num[i];
         ++i;
    }

    if(i!=0)
    average=sum/i;

    for(int j=0;j<i;++j)
    if(num[j]>average)
    ++total;

    cout<<"这些数字的平均值为"<<average<<endl;
    cout<<"并且共有"<<total<<"个数字大于平均值。\n";

    return 0;
}
```

```
#include <iostream>
using namespace std;
const int Num = 10;
int main()
{
    double donation[Num];
    int i = 0;
    int count =0;
    double sum = 0.0;
    cout << "Please enter: \n";
    while(i < Num && cin >> donation[i])
    {
        sum += donation[i++];
    }
    if(i == 0)
        cout << "No data--bye \n";
```

```
    else
    {
        double average = sum/i;
        for (int j = 0; j < i; j++)
        {
            if(donation[j] > average)
                ++count;
        }
        cout << "The average = "
             << average << endl
             << "The numbers bigger than the average: "
             << count << endl;
    }
    return 0;
}
```

//ex.6.2--array

```
#include <iostream>
#include<cctype>
#include<array>
int main()
{
    using namespace std;

    double sum=0,average=0;
     array<double,10>ad={0};
    int i=0,total=0;

    double temp;

    while(cin>>temp&&i<10&&!isdigit(temp))
    {
         ad[i]=temp;
         sum+=ad[i];
         ++i;
    }

    if(i!=0)
    average=sum/i;

    for(int j=0;j<i;++j)
    if(ad[j]>average)
    ++total;
```

```
    cout<<"这些数字的平均值为"<<average<<endl;
    cout<<"并且共有"<<total<<"个数字大于平均值。\n";

    return 0;
}
```

//ex.6.3

```
#include <iostream>

int main()
{
    using namespace std;

    cout<<"Please enter one of the following choices:\n"
        <<"c)carnivore          p)pianist\n"
        <<"t)tree               g)game\nf\n";    //书上的这个f个人认为是打印错误

    cout<<"Please enter a c, p, t, or g: ";

    char ch;
    cin>>ch;

    while(ch!='c'&&ch!='p'&&ch!='t'&&ch!='g')
    {
          cout<<"Please enter a c, p, t, or g: ";
          cin>>ch;
    }

    switch(ch)
    {
           case 'c':
                   cout<<"A maple is a carnivore.\n";
                   break;
           case 'p':
                   cout<<"A maple is a pianist.\n";
                   break;
           case 't':
                   cout<<"A maple is a tree.\n";
                   break;
           case 'g':
```

```
                cout<<"A maple is a game.\n";
    }
    return 0;
}

#include <iostream>
using namespace std;
void show();
int main()
{
    show();
    char choice;
    while (cin >> choice)
    {
        switch(choice)
        {
        case 'c' : cout << "It's a carnivore.\n";
                   break;
        case 'p' : cout << "It's a pianist.\n";
                   break;
        case 't' : cout << "A maple is a tree.\n";
                   break;
        case 'g' : cout << "It's a game.\n";
                   break;
        default  : cout << "Please enter a c, p, t, or g:";
        }
    }
    return 0;
}

void show()
{
    cout << "Please enter one of the following choices: \n"
           "c) carnivore          p) pianist\n"
           "t) tree               g) game\n";
}
```

//ex.6.4

```
#include <iostream>
const int strsize=20;

struct bop{
          char fullname[strsize];
          char title[strsize];
```

```
            char bopname[strsize];
            int preference;
};
int main()
{
    using namespace std;
    cout<<"Benevolent Order of Programmers Report\n"
        <<"a. display by name        b. display by title\n"
        <<"c. display by bopname     d. diplay by preference\n"
        <<"q. quit\n";
    char ch;
    bop member[5]={
                    {"Wimp Macho","English Teacher","DEMON",0},
                   {"Raki Rhodes","Junior Programmer","BOOM",1},
                   {"Celia Laiter","Super Star","MIPS",2},
                   {"Hoppy Hipman","Analyst Trainee","WATEE",1},
                   {"Pat Hand","Police","LOOPY",2}
    };
    cout<<"Enter your choice:";
    while(cin>>ch&&ch!='q')
    {
    switch(ch)
    {
          case 'a':
                   for(int i=0;i<5;i++)
                   cout<<member[i].fullname<<endl;
                   break;
          case 'b':
                   for(int i=0;i<5;i++)
                   cout<<member[i].title<<endl;
                   break;
          case 'c':
                   for(int i=0;i<5;i++)
                   cout<<member[i].bopname<<endl;
                   break;
          case 'd':
                   for(int i=0;i<5;i++)
                   {
                       if(member[i].preference==0)
                       cout<<member[i].fullname<<endl;
                       else if(member[i].preference==1)
                       cout<<member[i].title<<endl;
                       else if(member[i].preference==2)
                       cout<<member[i].bopname<<endl;
```

```
                }
                  break;
    }
    cout<<"Next choice: ";
    }
    cout<<"Bye!\n";
    return 0;
}

#include <iostream>
using namespace std;
const int strsize = 30;
struct bop{
    char fullname[strsize];
    char title[strsize];
    char bopname[strsize];
    int preference;
};
void show();
int main()
{
    bop A[5] =
    {
        {"Wimp Macho", "Teacher", "HAHA", 0},
        {"Raki Rhodes", "Junior Programmer", "LIAR", 1},
        {"Celia", "engineer", "MIPS", 2},
        {"Hoppy Hipman", "Analyst Trainee", "WAHU", 1},
        {"Pat Hand", "Student", "LOOPY", 2}
    };
    cout << "Benevolent Order of Programmers Report\n";
    show();
    cout << "Enter your choice: ";
    char choice;
    cin >> choice;
    while (choice != 'q')
    {
        switch(choice)
        {
        case 'a' : cout << A[0].fullname << endl << A[1].fullname << endl
                        << A[2].fullname << endl << A[3].fullname << endl
                        << A[4].fullname << endl;
                   break;
        case 'b' : cout << A[0].title << endl << A[1].title << endl
                        << A[2].title << endl << A[3].title << endl
```

```
                    << A[4].title << endl;
                break;
        case 'c' : cout << A[0].bopname << endl << A[1].bopname << endl
                    << A[2].bopname << endl << A[3].bopname << endl
                    << A[4].bopname << endl;
                break;
        case 'd' : cout << A[0].fullname << endl << A[1].title << endl
                    << A[2].bopname << endl << A[3].title << endl
                    << A[4].bopname << endl;
                break;
        default  : cout << "That's not the proper choice.\n";
        }
        cout << "Next choice: ";
        cin >> choice;
    }
    cout << "Bye!\n";
    return 0;
}
void show()
{
    cout << "a. display by name     b. display by title\n"
         << "c. display by bopname   d. display by preference\n"
         << "q. quit\n";
}
```

//ex.6.5

```
#include <iostream>
int main()
{
    using namespace std;

    double income,revenue;


    cout<<"请输入你的收入：";
    while(cin>>income&&income>=0)
    {
          if(income<=5000)
          revenue=0.0;
          else if(income<=15000)
          revenue=0.1*(income-5000);
          else if(income<=35000)
          revenue=0.1*(15000-5000)+0.15*(income-15000);
          else
```

```
            revenue=0.1*(15000-5000)+0.15*(35000-15000)+0.2*(income-35000);

            cout<<"你的所得税为"<<revenue<<endl;

            cout<<"请输入你的收入：";
    }
    return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
    double income,tax;
    cout << "Please enter your income: ";

    while (cin >> income && income >= 0)
    {
        if(income <= 5000)
            tax = 0;
        else if(income <= 15000)
            tax = 0.1*(income - 5000);
        else if(income <= 35000)
            tax = 10000*0.1 + 0.15*(income - 15000);
        else
            tax = 10000*0.1 + 0.15*20000 + 0.2*(income - 35000);
        cout << "Your tax is: " << tax << endl;
        cout << "Please enter your income: ";
    }
    cout << "Bye!\n";
    return 0;
}
```

```
//ex.6.6
#include <iostream>
#include <string>
using namespace std;
struct patron{
       string name;
       double money;
};

int main()
{
```

```
    int num,temp=0;

    cout<<"请输入捐款的人数：";
    cin>>num;
    cin.get();

    patron *ps=new patron[num];

    for(int i=0;i<num;++i)
    {
        cout<<"请输入第"<<i+1<<"位捐款人的名字：";
        getline(cin,ps[i].name);
        cout<<"请输入第"<<i+1<<"位捐款人捐款的数目：";
        cin>>ps[i].money;
        cin.get();
    }

    cout<<"Grand Patrons:\n";
    for(int i=0;i<num;++i)
    if(ps[i].money>10000)
    {
        cout<<ps[i].name<<"\n"<<ps[i].money<<endl;
        ++temp;
    }

    if(temp==0)
    cout<<"none\n";

    cout<<"Patrons:\n";
    for(int i=0;i<num;++i)
    if(ps[i].money<=10000)
    {
        cout<<ps[i].name<<"\n"<<ps[i].money<<endl;
        ++temp;
    }

    if(temp==0)
    cout<<"none\n";

    delete [] ps;
    return 0;
}

#include <iostream>
```

```cpp
#include <string>
using namespace std;
struct charity
{
    string name;
    double money;
};
int main()
{
    int number;
    int count = 0;
    cout << "Please enter the number of donator: ";
    cin >> number;
    charity *pt = new charity[number];
    for (int i = 0; i < number; i++)
    {
        cout << "Please enter your name: ";
        cin.get();
        getline(cin, pt[i].name);
        cout << "Please enter the money you are going to donate: ";
        cin >> pt[i].money;
       if(pt[i].money > 10000)
            count++;
    }
    if(count == 0)
        cout << "None(money > 10000)";
    else
    {
        cout << "Grand Patron\n";
        for(int i = 0; i < number; i++)
        {
            if(pt[i].money > 10000)
                cout << pt[i].name << " " << pt[i].money << endl;
        }
    }
    cout << endl;
    if(10 - count == 0)
        cout << "None(money < 10000)";
    else
    {
        cout << "Patron\n";
        for(int i = 0; i < number; i++)
        {
            if(pt[i].money < 10000)
```

```
                cout << pt[i].name << " " << pt[i].money << endl;
        }
    }
    return 0;
}
```

//ex.6.7

```
#include <iostream>
#include <cctype>

int main()
{
    using namespace std;

    int vowel=0,consonant=0,other=0;
    char word[15];

    cout<<"Enter words (q to quit):\n";
    while(cin>>word)
    {
          if(isalpha(word[0]))
          {
              if(word[0]=='q'&&strlen(word)==1)
              break;
              else if(word[0]=='a'||word[0]=='i'||
              word[0]=='u'||word[0]=='e'||word[0]=='o')
              ++vowel;
              else
              ++consonant;
           }
           else
           ++other;
    }

    cout<<vowel<<" words beginning with vowels\n";
    cout<<consonant<<" words beginning with consonants\n";
    cout<<other<<" others\n";
    return 0;
}
```

```
#include <iostream>
#include <cctype>
using namespace std;
int main()
```

```
{
    char word[20];
    int vow = 0, consonant = 0, other =0;
    cout << "Enter words (q to quit):\n";
    while(cin >> word)
    {
        if (isalpha(word[0]))
        {
            if(word[0] == 'a' || word[0] == 'e' || word[0] == 'i'|| word[0]
== 'o' || word[0] == 'u' ||
            word[0] == 'A' || word[0] == 'E' || word[0] == 'I' || word[0] ==
'O' || word[0] == 'U')
                vow++;
            else if(word[0] == 'q' && strlen(word) == 1)
                break;
            else
                consonant++;
        }
        else
            other++;
    }
    cout << vow << " words beginning with vowels\n"
         << consonant << " words beginning with consonants\n"
         << other << " others\n";
    return 0;
}
```

//ex.6.8

```
#include <iostream>
#include <fstream>
#include <cstdlib>

int main()
{
    using namespace std;

    char ch;
    int sum=0;

    ifstream inFile;
    inFile.open("abc.txt");
    if(!inFile.is_open())
    {
        cout<<"Could not open the file \n";
```

```
        cout<<"Program terminating.\n";
        exit(EXIT_FAILURE);
    }

    inFile>>ch;
    while(inFile.good())
    {
        ++sum;
        inFile>>ch;
    }

    if(inFile.eof())
    cout<<"End of file reached.\n";
    else if(inFile.fail())
    cout<<"Input terminated by data mismatch.\n";
    else
    cout<<"Input terminated for unkonwn reason.\n";

    cout<<"总共有"<<sum<<"个字符在这个文件中。"<<endl;
    return 0;
}

#include <iostream>
#include <fstream>
#include <cstdlib>
using namespace std;
const int Size = 20;
int main()
{
    char filename[Size];
    ifstream infile;
    cout << "Enter name of data file: ";
    cin.getline(filename, Size);
    infile.open(filename);
    if (!infile.is_open())
    {
        cout << "Could not open the file " <<filename << endl;
        cout << "Program terminating.\n";
        exit(EXIT_FAILURE);
    }
    char a;
    int count = 0;

    infile >> a;
```

```
    while (infile.good())
    {
        ++count;
        infile >> a;
    }
    if (infile.eof())
        cout << "End of file reached.\n";
    else if (infile.fail())
        cout << "Input terminated by data mismatch.\n";
    else
        cout << "Input terminated for unknown reason.\n";
    if (count == 0)
        cout << "No data processed.\n";
    else
        cout << "The text contains " << count << " character(s)" << endl;

    infile.close();
    return 0;
}
```

//ex.6.9

```
#include <iostream>
#include <fstream>
#include <cstdlib>

struct member
{
       char name[20];
       double donation;
};

int main()
{
    using namespace std;
    int num,count1=0,count2=0;
    ifstream fin;
    char file[20];
    cout<<"Enter name of data file: ";
    cin.getline(file,20);
    fin.open(file);
    if(!fin.is_open())
    {
        cout<<"Could not open the file-"<<file<<endl;
        cout<<"Program terminating.\n";
```

```
        exit(EXIT_FAILURE);
    }
    fin>>num;
    fin.get();
    member *pd=new member[num];
    for(int i=0;i<num;i++)
    {
        fin.getline(pd[i].name,20);
        fin>>pd[i].donation;
        fin.get();
    }
    cout<<"Grand Patrons:\n";
    for(int i=0;i<num;i++)
    if(pd[i].donation>=10000)
    {
        cout<<pd[i].name<<"\n"<<pd[i].donation<<endl;
        count1++;
    }
    if(count1==0)
    cout<<"none\n";
    cout<<"Patrons:\n";
    for(int i=0;i<num;i++)
    if(pd[i].donation<10000)
    {
        cout<<pd[i].name<<"\n"<<pd[i].donation<<endl;
        count2++;
    }
    if(count2==0)
    cout<<"none\n";

    delete [] pd;
    return 0;
}

#include <iostream>
#include <fstream>
#include <cstdlib>
#include <string>
using namespace std;
struct charity
{
    string name;
    double money;
};
```

```
int main()
{
    string filename;
    ifstream infile;
    cout << "Enter name of data file: ";
    getline(cin, filename);
    infile.open(filename);
    if (!infile.is_open())
    {
        cout << "Could not open the file " <<filename << endl;
        cout << "Program terminating.\n";
        exit(EXIT_FAILURE);
    }

    int number,count = 0;
    infile >> number;
    charity *pt = new charity[number];
    for (int i = 0; i < number; i++)
    {
        infile.get();
        getline(infile, pt[i].name);
        infile >> pt[i].money;
       if(pt[i].money > 10000)
            count++;
    }
    if(count == 0)
        cout << "None(money > 10000)";
    else
    {
        cout << "Grand Patron:\n";
        for(int i = 0; i < number; i++)
        {
            if(pt[i].money > 10000)
                cout << pt[i].name << " " << pt[i].money << endl;
        }
    }
    if(10 - count == 0)
        cout << "None(money < 10000)";
    else
    {
        cout << "Patron:\n";
        for(int i = 0; i < number; i++)
        {
            if(pt[i].money < 10000)
```

```
                cout << pt[i].name << " " << pt[i].money << endl;
        }
    }
    delete [] pt;
    return 0;
}
```

第 7 章　函数——C++的编程模块

//ex7.1

```
#include <iostream>
double t_av(double x, double y);
int main()
{
using namespace std;
double x, y;
double result;
cout << "Please enter two numbers (0 to stop): ";
while ((cin >> x >> y) && x != 0 && y != 0)
{
result = t_av(x, y);
cout << "调和平均数 = " << result << endl;
cout << "Please enter two numbers (0 to stop): ";
}
return 0;
}
double t_av(double x, double y)
{
return 2.0 * x * y / (x + y);
}
```

```
#include <iostream>
using namespace std;
void average(double, double);
int main()
```

```
{
    double A, B;
    cout << "Please enter two numbers: ";
    while (cin >> A >> B)
    {
        if (A == 0 || B == 0)
            break;
        else
            average(A, B);
        cout << "Please enter two numbers: ";
    }
    cout << "Bye!\n";
    return 0;
}
void average(double x, double y)
{
    cout << "The average is: "
         << 2.0 * x * y / (x + y)
         << endl;
}

//ex7.2
#include <iostream>
const int MAX = 10;
using namespace std;
int fill_ar(double ar[], int limit);
void show_ar(const double ar[], int n);
double average(const double ar[], int n);
int main()
{
double scores[MAX];
int size = fill_ar(scores, MAX);
show_ar(scores, size);
if (size > 0)
cout << "The average of scores is: "
<< average(scores, size) << endl;
return 0;
}
int fill_ar(double ar[], int limit)
{
double temp;
int i;
for (i = 0; i < limit; i++)
{
```

```
cout << "Enter score #" << i+1 << ": ";
cin >> temp;
if (!cin)
{
cin.clear();
while (cin.get() != '\n')
continue;
cout << "Bad input; enter a number: ";
break;
}
if (temp < 0)
break;
ar[i] = temp;
}
return i;
}
void show_ar(const double ar[], int n)
{
for (int i = 0; i < n; i++)
cout << "score #" << i+1 << ": " << ar[i] << endl;
}
double average(const double ar[], int n)
{
double sum = 0.0;
for (int i = 0; i < n; i++)
sum += ar[i];
return sum / n;
}

#include <iostream>
using namespace std;
double score[10];
int input(double [], int);
void average(double [], int);
void show(const double[], int);
int main()
{
    int size = input(score, 10);
    if (size > 0)
    {
        show(score, size);
        average(score, size);
    }
    cout << "Done.\n";
```

```
    return 0;
}
int input(double score[], int limit)
{
    double a;
    int i;
    for (i = 0; i < limit; i++)
    {
        cout << "Your score: ";
        cin >> a;
        if (!cin)
        {
            cin.clear();
            while (cin.get() != '\n')
                continue;
            cout << "Bad input; input process terminated.\n";
            break;
        }
        else if (a < 0)
            break;
        score[i] = a;

    }
    return i;
}
void show(const double ar[], int n)
{
    double total = 0.0;
    cout << "Score: ";
    for (int i = 0; i < n; i++)
    {
        cout << ar[i] << "  ";
    }
    cout << endl;
}
void average(double ar[], int n)
{
    double av,total = 0.0;
    int i;
    for (i = 0; i < n; i++)
    {
        total += ar[i];
    }
    av = total / i;
```

```
    cout << "The average score: " << av << endl;
}

//ex7.3
#include <iostream>
struct box
{
char maker[40];
float height;
float width;
float length;
float volume;
};
void set_box(box *);
void show_box(box);
int main()
{
using namespace std;
box carton = {"Bingo Boxer", 2, 3, 5};
set_box(&carton);
show_box(carton);
return 0;
}
void set_box(box * pb)
{
pb->volume = pb->height * pb->length * pb->width;
}
void show_box(box b)
{
using namespace std;
cout << "Box maker: " << b.maker
<< "\nheight: " << b.height
<< "\nlwidth: " << b.width
<< "\nlength: " << b.length
<< "\nvolume: " << b.volume << endl;
}

#include <iostream>
using namespace std;
struct box
{
    char maker[40];
    float height;
    float width;
```

```
    float length;
    float volume;
};
void show(box);
box* calculate(box *);
int main()
{
    box a = {"M", 3.4, 4.5, 5.6, 0.0};
    show(a);
    box* pt;
    cout << endl;
    pt = calculate(&a);
    show(a);
    return 0;
}
void show(box x)
{
    cout << "The height of the box: " << x.height << endl
         << "The width of the box: " << x.width << endl
         << "The length of the box: " << x.length << endl
         << "The volume of the box: " << x.volume << endl;
}
box* calculate(box* ps)
{
    (*ps).volume = (*ps).height * (*ps).length * (*ps).width;
    return ps;
}

//ex7.4
#include <iostream>
long double probability(unsigned numbers, unsigned picks);
int main()
{
using namespace std;
double total, choices, mtotal;
long double probability1, probability2;
cout << "Enter total number of game card choices and\n"
"number of picks allowed for the field:\n";
while ((cin >> total >> choices) && choices < total)
{
cout << "Enter total number of game card choices and\n"
"number of picks allowed for the mega:\n";
if (!(cin >> mtotal))
break;
```

```
probability1 = probability(total, choices);
probability2 = probability(mtotal, 1);
cout << "The chances of getting all " << choices << " picks is
one in "
<< probability1 << ".\n";
cout << "The chances of getting the megaspot is one in "
<< probability2 << ".\n";
cout << "You have one chance in ";
cout << probability1 * probability2;
cout << " of winning.\n";
cout << "Next set of numbers (q to quit): ";
}
cout << "bye\n";
return 0;
}
long double probability(unsigned numbers, unsigned picks)
{
long double result = 1.0;
long double n;
unsigned p;
for (n = numbers, p = picks; p > 0; n--, p--)
result = result * n / p;
return result;
}

#include <iostream>
using namespace std;
long double probability1(unsigned, unsigned);
long double probability2(unsigned);
int main()
{
    double a, b, c;
    cout << "Enter the number of choices on the game card and\n"
         << "the number of picks allowed(in 1 field number):\n";
    cin >> a >> b;
    cout << "Enter the number of choices on the game card \n"
         << "(in 2 field number):\n";
    cin >> c;
    while (cin)
    {
       if (b <= a)
       {
           long double chance;
           chance = probability1(a, b) * probability2(c);
```

```
            cout << "You have one chance in "
                 << chance << " of wining.\n\n";
            cout << "Enter the number of choices on the game card and\n"
                 << "the number of picks allowed(in 1 field number):\n";
            cin >> a >> b;
            if (!cin)
                break;
            cout << "Enter the number of choices on the game card \n"
                 << "(in 2 field number):\n";
            cin >> c;
        }
        else
            break;
    }
    cout << "Bye!\n";
    return 0;
}
long double probability1(unsigned numbers, unsigned picks)
{
    long double result = 1.0;
    long double n;
    unsigned p;
    for (n = numbers, p = picks; p > 0; n--, p--)
        result = result * n / p;
    return result;
}
long double probability2(unsigned numbers)
{
    long double result;
    result = 1.0 / numbers;
    return result;
}
```

//ex7.5

```
#include <iostream>
long long int recure(int);
int main()
{
using namespace std;
int number;
cout << "Enter a integer (q to stop): ";
```

```
while (cin >> number)
{
long long int result = recure(number);
cout << number << "! = " << result << endl;
cout << "Next:";
}
cout << "Done!" << endl;
return 0;
}
long long int recure(int n)
{
long long int result;
if (n > 0)
result = n * recure(n-1);
else
result = 1;
return result;
}
```

```
#include <iostream>
using namespace std;
unsigned long sub(int);
int main()
{
    cout << "Enter one integer: (q to quit)";
    int num;
    while(cin >> num)
    {
        unsigned long result = sub(num);
        cout << "The result of " << num << "! is: "
             << result << endl
             << "Next number: ";
    }
    return 0;
}
unsigned long sub(int n)
{
    unsigned long result = n;
    if (result > 0)
        result = result * sub(n - 1);
    else
        result = 1;
    return result;
}
```

```
//ex7.6
#include <iostream>
const int Size = 10;
int Fill_array(double ar[], int n);
void Show_array(const double ar[], int n);
void Reverse_array(double ar[], int n);
int main()
{
using namespace std;
double values[Size];
int len = Fill_array(values, Size);
cout << "Array values:\n";
Show_array(values, len);
cout << "Array reversed:\n";
Reverse_array(values, len);
Show_array(values, len);
cout << "All but end values reversed:\n";
Reverse_array(values+1, len-2);
Show_array(values, len);
return 0;
}
int Fill_array(double ar[], int n)
{
using namespace std;
double temp;
int i;
for (i=0; i<n; i++)
{
cout << "Enter value #" << i+1 << ": ";
cin >> temp;
if (!cin)
break;
ar[i] = temp;
}
cout << endl;
return i;
}
void Show_array(const double ar[], int n)
{
using namespace std;
for (int i=0; i<n; i++)
cout << "Property #" << i+1 << ": "
<< ar[i] << endl;
```

```
cout << endl;
}
void Reverse_array(double ar[], int n)
{
double temp;
for (int i=0,j=n-1; i<j; i++,j--)
{
temp = ar[i];
ar[i] = ar[j];
ar[j] = temp;
}
}

#include <iostream>
using namespace std;
const int Asize = 10;
int Fill_array(double [], int);
void Show_array(double [], int);
double * Reverse_array(double [], int, int);
int main()
{
    double numbers[Asize];
    cout << "Please enter some numbers(less than ten): \n";
    int i = Fill_array(numbers, Asize);
        cout << "You've entered " << i << " numbers:\n";
    Show_array(numbers, i);
    cout << endl;
    double * pt = Reverse_array(numbers, 0, i);
    Show_array(pt, i);
    cout << endl;
    double * ps = Reverse_array(numbers, 1, i);
    Show_array(ps, i);
    cout << endl;
    return 0;
}
int Fill_array(double ar[], int size)
{
    int i;
    for (i = 0; i < Asize; i++)
    {
        if (cin >> ar[i])
            ;
        else
            break;
```

```
    }
    return i;
}
void Show_array(double ar[], int size)
{
    for (int i = 0; i < size; i++)
        cout << ar[i] << "  ";
}
double * Reverse_array(double ar[], int a, int size)
{
    cout << "Here are(is) the number(s) after reverse:\n";
    double temp;
    for (int i = a; i < size / 2; i++)
    {
        temp = ar[i];
        ar[i] = ar[size - i - 1];
        ar[size - i - 1] = temp;
    }
    return ar;
}

//ex7.7
#include <iostream>
const int Max = 5;
double * fill_array(double * begin, double * end);
void show_array(const double * begin, const double * end);
void revalue(double r, double * begin, double * end);
int main()
{
using namespace std;
double properties[Max];
double * pbegin = properties;
double * pend = fill_array(pbegin, pbegin + Max);
show_array(pbegin, pend);
if (pend-pbegin > 0)
{
cout << "Enter revaluation factor: ";
double factor;
while (!(cin >> factor))
{
cin.clear();
while (cin.get() != '\n')
continue;
cout << "Bad input; Please enter a number: ";
```

```
}
revalue(factor, pbegin, pend);
show_array(pbegin, pend);
}
cout << "Done.\n";
return 0;
}
double * fill_array(double * begin, double * end)
{
using namespace std;
double temp;
int i = 1;
while (begin < end)
{
cout << "Enter value #" << i << ": ";
cin >> temp;
if (!cin)
{
cin.clear();
while (cin.get() != '\n')
continue;
cout << "Bad input; input process terminated.\n";
break;
}
else if (temp < 0)
break;
*begin = temp;
begin++;
i++;
}
return begin;
}
void show_array(const double * begin, const double * end)
{
using namespace std;
int i = 1;
while (begin < end)
{
cout << "Property #" << i << ": $";
cout << *begin << endl;
begin++;
i++;
}
}
```

```
void revalue(double r, double * begin, double * end)
{
while (begin < end)
{
*begin *= r;
begin++;
}
}

#include <iostream>
using namespace std;
const int Max = 5;
void show_array(const double [], double *);
void revalue(double, double [], double *);
double * fill_array(double [], int);
int main()
{
    double properties[Max];
    double * p = fill_array(properties, Max);
    show_array(properties, p);
    if (p != &properties[0])
    {
        cout << "Enter revaluation factor: ";
        double factor;
        while (!(cin >> factor))
        {
            cin.clear();
            while (cin.get() != '\n')
                continue;
            cout << "Bad input; Please enter a number: ";
        }
        revalue(factor, properties, p);
        show_array(properties, p);
    }
    cout << "Done.\n";
    return 0;
}
double * fill_array(double ar[], int n)
{
    double temp;
    int i;
    for (i = 0; i < n; i++)
    {
        cout << "Enter value #" << (i + 1) << ": ";
```

```
        cin >> temp;
        if (!cin)
        {
            cin.clear();
            while (cin.get() != '\n')
                continue;
            cout << "Bad input; input process terminated.\n";
            break;
        }
        else if (temp < 0)
            break;
        ar[i] = temp;
    }
    double * pt = &ar[i - 1];
    return pt;
}
void show_array(const double ar[], double * ps)
{
    const double * p = &ar[0];
    for (int i = 0; p != ps + 1; p++,i++)
    {
        cout << "Property #" << (i + 1) << ": $";
        cout << ar[i] << endl;
    }
}
void revalue(double r, double ar[], double * ps)
{
    double * p = &ar[0];
    for (int i = 0; p != ps + 1; p++, i++)
        ar[i] *= r;
}


//ex7.8a
#include <iostream>
const int Seasons = 4;
const char * Snames[] = {"Spring", "Summer", "Fall", "Winter"};
void fill(double ar[], int n);
void show(double ar[], int n);
int main()
{
using namespace std;
double expenses[Seasons];
fill(expenses, Seasons);
```

```
show(expenses, Seasons);
return 0;
}
void fill(double ar[], int n)
{
using namespace std;
for (int i=0; i<n; i++)
{
cout << "Enter " << Snames[i] << " expenses: ";
cin >> ar[i];
}
}
void show(double ar[], int n)
{
using namespace std;
cout << "\nEXPENSES\n";
double total = 0.0;
for (int i=0; i<n; i++)
{
cout << Snames[i] << ": $" << ar[i] <<endl;
total += ar[i];
}
cout << "Total Expenses: $" << total << endl;
}
```

```
#include <iostream>
using namespace std;
const char * Seasons[4] = {"Spring", "Summer", "Fall", "Winter"};
void fill(double *);
void show(double []);
int main()
{
    double expenses[4];
    fill(expenses);
    show(expenses);
    return 0;
}
void fill(double ar[])
{
    double costs;
    for (int i = 0; i < 4; i++)
    {
        cout << "Enter " << Seasons[i] << " expenses: ";
        cin >> ar[i];
```

```
    }
}
void show(double ar[])
{
    double total = 0.0;
    cout << "\nEXPENSES\n";
    for (int i = 0; i < 4; i++)
    {
        cout << Seasons[i] << ": $" << ar[i] << endl;;
        total += ar[i];
    }
    cout << "Total Expenses: $" << total << endl;
}

//ex7.8b（传递结构值）
#include <iostream>
const int Seasons = 4;
struct data
{
double arr[Seasons];
};
const char * Snames[] = {"Spring", "Summer", "Fall", "Winter"};
data fill();
void show(data);
int main()
{
using namespace std;
data expenses = fill();
show(expenses);
return 0;
}
data fill()
{
using namespace std;
data expenses;
for (int i=0; i<Seasons; i++)
{
cout << "Enter " << Snames[i] << " expenses: ";
cin >> expenses.arr[i];
}
return expenses;
}
void show(data expenses)
{
```

```
using namespace std;
cout << "\nEXPENSES\n";
double total = 0.0;
for (int i=0; i<Seasons; i++)
{
cout << Snames[i] << ": $" << expenses.arr[i] <<endl;
total += expenses.arr[i];
}
cout << "Total Expenses: $" << total << endl;
}
//ex7.8b（传递结构指针）
#include <iostream>
const int Seasons = 4;
struct data
{
double arr[Seasons];
};
const char * Snames[] = {"Spring", "Summer", "Fall", "Winter"};
void fill(data * pd);
void show(data * pd);
int main()
{
using namespace std;
data expenses;
fill(&expenses);
show(&expenses);
return 0;
}
void fill(data * pd)
{
using namespace std;
for (int i=0; i<Seasons; i++)
{
cout << "Enter " << Snames[i] << " expenses: ";
cin >> pd->arr[i];
}
}
void show(data * pd)
{
using namespace std;
cout << "\nEXPENSES\n";
double total = 0.0;
for (int i=0; i<Seasons; i++)
{
```

```
cout << Snames[i] << ": $" << pd->arr[i] <<endl;
total += pd->arr[i];
}
cout << "Total Expenses: $" << total << endl;
}

#include <iostream>
using namespace std;
const char * Seasons[4] = {"Spring", "Summer", "Fall", "Winter"};
struct expenditure
{
    double expenses[4];
};
expenditure fill(expenditure);
void show(expenditure);
int main()
{
    expenditure a = {{0.0}};
    expenditure v = fill(a);
    show(v);
    return 0;
}
expenditure fill(expenditure b)
{
    for (int i = 0; i < 4; i++)
    {
        cout << "Enter " << Seasons[i] << " expenses: ";
        cin >> b.expenses[i];
    }
    return b;
}
void show(expenditure b)
{
    double total = 0.0;
    cout << "\nEXPENSES\n";
    for (int i = 0; i < 4; i++)
    {
        cout << Seasons[i] << ": $" << b.expenses[i] << endl;
        total += b.expenses[i];
    }
    cout << "Total Expenses: $" << total << endl;
}
```

//ex7.9

```
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
using namespace std;
const int SLEN = 30;
struct student {
char fullname[SLEN];
char hobby[SLEN];
int ooplevel;
};
int getinfo(student pa[], int n);
void display1(student st);
void display2(const student * ps);
void display3(const student pa[], int n);
int main()
{
cout << "Enter class size: ";
int class_size;
cin >> class_size;
while (cin.get() != '\n')
continue;
student * ptr_stu = new student[class_size];
int entered = getinfo(ptr_stu, class_size);
for (int i = 0; i < entered; i++)
{
display1(ptr_stu[i]);
display2(&ptr_stu[i]);
}
display3(ptr_stu, entered);
delete [] ptr_stu;
cout << "Done\n";
return 0;
}
// getinfo() has two arguments: a pointer to the first element of
// an array of student structures and an int representing the
// number of elements of the array. The function solicits and
// stores data about students. It terminates input upon filling
// the array or upon encountering a blank line for the student
// name. The function returns the actual number of array elements
// filled.
int getinfo(student pa[], int n)
```

```
{
int num_array_elem = n;
char tmp[SLEN];
for (int i = 0; i < n; ++i)
{
cout << "Enter name: ";
cin.getline(tmp, SLEN);
bool blank_line = true;
for (unsigned j = 0; j < strlen(tmp); ++j)
{
if (!isspace(tmp[j]))
{
blank_line = false;
break;
}
}
if (blank_line)
{
num_array_elem = i;
break;
}
strcpy(pa[i].fullname, tmp);
cout << "Enter hobby: ";
cin.getline(pa[i].hobby, SLEN);
cout << "Enter ooplevel: ";
cin >> pa[i].ooplevel;
cin.get();
}
cout << endl;
return num_array_elem;
}
// display1() takes a student structure as an argument
// and displays its contents
void display1(student st)
{
cout << st.fullname << '\t' << st.hobby << '\t' << st.ooplevel <<
endl;
}
// display2() takes the address of student structure as an
// argument and displays the structure'¡¥s contents
void display2(const student * ps)
{
cout << ps->fullname << '\t' << ps->hobby << '\t' << ps->ooplevel
<< endl;
```

```
}
// display3() takes the address of the first element of an array
// of student structures and the number of array elements as
// arguments and displays the contents of the structures
void display3(const student pa[], int n)
{
for (int i = 0; i < n; ++i)
cout << pa[i].fullname << '\t' << pa[i].hobby << '\t' <<
pa[i].ooplevel << endl;
}

#include <iostream>
using namespace std;
const int SLEN = 30;
struct student {
    char fullname[SLEN];
    char hobby[SLEN];
    int ooplevel;
};
int getinfo(student pa[], int n);
void display1(student st);
void display2(const student * ps);
void display3(const student pa[], int n);
int main()
{
    cout << "Enter class size: ";
    int class_size;
    cin >> class_size;
    while (cin.get() != '\n')
        continue;
    student * ptr_stu = new student[class_size];
    int entered = getinfo(ptr_stu, class_size);
    for (int i = 0; i < entered; i++)
    {
        display1(ptr_stu[i]);
        display2(&ptr_stu[i]);
    }
    display3(ptr_stu, entered);
    delete [] ptr_stu;
    cout << "Done\n";
    return 0;
}
int getinfo(student * p, int num)
{
```

```
    int i;
    for (i = 0; i < num; i++)
    {
        cout << "Enter the fullname: ";
        cin.getline((p + i)->fullname, SLEN);
        cout << "Enter the hobby: ";
        cin.getline((p + i)->hobby, SLEN);
        cout << "Enter the ooplevel: ";
        cin >> (p + i)->ooplevel;
        if (!cin)
            break;
        else
            cin.get();
    }
    return i;
}
void display1(student st)
{
    cout << st.fullname << " "
         << st.hobby << " "
         << st.ooplevel << endl;
}
void display2(const student * ps)
{
    cout << ps->fullname << " "
         << ps->hobby << " "
         << ps->ooplevel << endl;
}
void display3(const student pa[], int num)
{
    for (int i = 0; i < num; i++)
    {
        cout << pa[i].fullname << " "
             << pa[i].hobby << " "
             << pa[i].ooplevel << endl;
    }
}

//ex7.10
#include <iostream>
double calculate(double x, double y, double (*pf)(double, double));
double add(double x, double y);
double sub(double x, double y);
double mean(double x, double y);
```

```
int main()
{
using namespace std;
double a, b;
double (*pf[3])(double, double) = {add, sub, mean};
char * op[3] = {"add", "sub", "mean"};
cout << "Enter pairs of numbers (q to quit): ";
while (cin >> a >> b)
{
for (int i=0; i<3; i++)
{
cout << op[i] << ": " << a << " and " << b << " = "
<< calculate(a, b, pf[i]) << endl;
}
}
}
double calculate(double x, double y, double (*pf)(double, double))
{
return (*pf)(x, y);
}
double add(double x, double y)
{
return x + y;
}
double sub(double x, double y)
{
return x - y;
}
double mean(double x, double y)
{
return (x + y) / 2.0;
}
```

第 8 章 函数探幽

//ex8.1

```
#include <iostream>
void show(const char * ps, int n = 0);
int main()
{
```

```
using namespace std;
char * pstr = "Hello\n";
show(pstr);
int num;
cout << "Enter a number: ";
cin >> num;
show(pstr, num);
cout << "Done\n";
return 0;
}
void show(const char * ps, int n)
{
using namespace std;
int lim = n;
if (n == 0)
lim = 1;
for (int i=0; i<lim; i++)
cout << ps;
}
//ex8.2
#include <iostream>
#include <string>
using namespace std;
struct CandyBar
{
string name;
double weight;
int hot;
};
void set(CandyBar & cb, char * ps, double w, int h);
void show(const CandyBar & cb);
int main()
{
using namespace std;
CandyBar candy;
char * p = "Millennium Munch";
double x = 2.85;
int y = 350;
set(candy, p, x, y);
show(candy);
return 0;
}
void set(CandyBar & cb, char * ps, double w, int h)
{
```

```
cb.name = ps;
cb.weight = w;
cb.hot = h;
}
void show(const CandyBar & cb)
{
cout << "Name: " << cb.name << endl
<< "Weight: " << cb.weight << endl
<< "Hot: " << cb.hot << endl;
}

#include <iostream>
#include <string>
using namespace std;
struct CandyBar {
    string brand;
    double weight;
    int calories;
};
void function(CandyBar &, char * str = "Millennium Munch", double a = 2.85,
int b = 350);
void show(const CandyBar &);
int main()
{
    CandyBar x;
    cout << "Please enter the brand: \n";
    getline(cin, x.brand);
    cout << "Please enter the weight: \n";
    cin >> x.weight;
    cout << "Please enter the colories: \n";
    cin >> x.calories;
    show(x);
    function(x);
    show(x);
    return 0;
}
void show(const CandyBar &a)
{
    cout << endl << a.brand << endl
         << a.weight << endl
         << a.calories << endl;
}
void function(CandyBar &r, char * str, double a, int b)
{
```

```
    r.brand = str;
    r.weight = a;
    r.calories = b;
}

//ex8.3
#include <iostream>
#include <string>
#include <cctype>
using namespace std;
void str_to_upper(string & str);
int main()
{
string str1;
cout << "Enter a string (q to quit): ";
while (getline(cin, str1) && str1!="q" && str1!="Q")
{
str_to_upper(str1);
cout << str1 << endl;
cout << "Next string (q to quit): ";
}
cout << "Bye.";
return 0;
}
void str_to_upper(string & str)
{
int limit = str.size();
for (int i=0; i<limit; i++)
{
if (isalpha(str[i]))
str[i] = toupper(str[i]);
}
}

#include <iostream>
#include <string>
#include <cctype>
using namespace std;
void upper(string &);
int main()
{
    string str;
    cout << "Enter a string (q to quit): ";
    getline(cin, str);
```

```
    while (str != "q" && str != "Q")
    {
        upper(str);
        cout << str << endl;
        cout << "Next string (q to quit): ";
        getline(cin, str);
    }
    cout << "Bye.\n";
    return 0;
}
void upper(string & a)
{
    for (int i = 0; i < a.size(); i++)
    {
        if (islower(a[i]))
            a[i] = toupper(a[i]);
    }
}

// ex8.4
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
#include <cstring> // for strlen(), strcpy()
using namespace std;
struct stringy {
char * str; // points to a string
int ct; // length of string (not counting '\0')
};
void show(const char *str, int cnt = 1);
void show(const stringy & bny, int cnt = 1);
void set(stringy & bny, const char * str);
int main(void)
{
stringy beany;
char testing[] = "Reality isn't what it used to be.";
set(beany, testing); // first argument is a reference,
// allocates space to hold copy of testing,
// sets str member of beany to point to the
// new block, copies testing to new block,
// and sets ct member of beany
show(beany); // prints member string once
show(beany, 2); // prints member string twice
testing[0] = 'D';
testing[1] = 'u';
```

```
show(testing); // prints testing string once
show(testing, 3); // prints testing string thrice
show("Done!");
return 0;
}
void show(const char *str, int cnt)
{
while(cnt-- > 0)
{
cout << str << endl;
}
}
void show(const stringy & bny, int cnt)
{
while(cnt-- > 0)
{
cout << bny.str << endl;
}
}
void set(stringy & bny, const char * str)
{
bny.ct = strlen(str);
bny.str = new char[bny.ct+1];
strcpy(bny.str, str);
}

#include <iostream>
using namespace std;
#include <cstring>
struct stringy {
    char * str;
    int ct;
};
void set(stringy &, char []);
void show(const stringy &, int n = 1);
void show(const char [], int n = 1);
int main()
{
    stringy beany;
    char testing[] = "Reality isn't what it used to be.";
    set(beany, testing);
    show(beany);
    show(beany, 2);
    testing[0] = 'D';
```

```
    testing[1] = 'u';
    show(testing);
    show(testing, 3);
    show("Done!");
    return 0;
}
void set(stringy & a, char b[])
{
    a.str = b;
}
void show(const stringy &x, int n)
{
    for (int i = n; i > 0; i--)
        cout << x.str << endl;
}
void show(const char a[], int n)
{
    for (int i = n; i > 0; i--)
        cout << a << endl;
}

//ex8.5
#include <iostream>
const int Limit = 5;
template <typename T>
T max5(T ar[]);
int main()
{
using namespace std;
int ari[Limit] = {1, 2, 3, 5, 4};
double ard[Limit] = {1.1, 2.2, 3.3, 5.5, 4.4};
int maxi = max5(ari);
double maxd = max5(ard);
cout << "maxi = " << maxi << endl;
cout << "maxd = " << maxd << endl;
return 0;
}
template <typename T>
T max5(T ar[])
{
T max = ar[0];
for (int i=1; i<Limit; i++)
{
if (max < ar[i])
```

```
max = ar[i];
}
return max;
}

#include <iostream>
using namespace std;
const int Num = 5;
template <class AnyType>
AnyType max5(AnyType []);
int main()
{
    int a[Num] = {1, 2, 3, 4, 5};
    double b[Num] = {1.1, 2.2, 3.3, 4.4, 5.5};
    int maxi = max5(a);
    double maxd = max5(b);
    cout << "max in a[5]: " << maxi << endl
         << "max in b[5]: " << maxd << endl;
    return 0;
}
template <class AnyType>
AnyType max5(AnyType ar[])
{
    AnyType max = ar[0];
    for (int i = 0; i < 5; i++)
    {
        if (max < ar[i])
            max = ar[i];
    }
    return max;
}

//ex8.6
#include <iostream>
template <typename T>
T maxn(T ar[], int n);
template <> const char* maxn(const char* ar[], int n);
int main()
{
using namespace std;
int ari[6] = {1, 2, 3, 4, 6, 5};
double ard[4] = {1.1, 2.2, 4.4, 3.3};
const char * ars[5] = {
"a",
```

```
"bb",
"ccc",
"ddddd",
"eeee"
};
cout << "The max integer of array is: " << maxn(ari, 6) << endl;
cout << "The max double of array is: " << maxn(ard, 4) << endl;
cout << "The max string of array is: " << maxn(ars, 5)<<endl;
}
template <typename T>
T maxn(T ar[], int n)
{
T maxar = ar[0];
for (int i=1; i<n; i++)
{
if (maxar < ar[i])
maxar = ar[i];
}
return maxar;
}
template <> const char* maxn(const char* ar[],int n)
{
const char * maxs = ar[0];
for (int i=1; i<n; i++)
{
if (strlen(maxs) < strlen(ar[i]))
maxs = ar[i];
}
return maxs;
}

#include <iostream>
using namespace std;
template <typename T>
T maxn(T [], int);
template <> const char* maxn<const char *>(const char * [], int);
int main()
{
    int a[6] = {1, 2, 3, 4, 5, 6};
    double b[4] = {1.1, 2.2, 3.3, 4.4};
    int maxi = maxn(a, 6);
    double maxd = maxn(b, 4);
    const char * c[5] = {
        "a",
```

```
        "bb",
        "ccc",
        "ddddd",
        "eeee"
    };
    cout << "maxi: " << maxi << endl
         << "maxd: " << maxd << endl
         << "The max string of array is: " << maxn(c, 5) << endl;
    return 0;
}
template <typename T>
T maxn(T ar[], int n)
{
    T max = ar[0];
    for (int i = 0; i < n; i++)
    {
        if (max < ar[i])
            max = ar[i];
    }
    return max;
}
template <> const char* maxn<const char *>(const char * ar[],int n)
{
const char * maxs = ar[0];
for (int i=1; i<n; i++)
{
if (strlen(maxs) < strlen(ar[i]))
maxs = ar[i];
}
return maxs;
}

//ex8.7
#include <iostream>
template <typename T>
T SumArrray(T arr[], int n);
template <typename T>
T SumArrray(T * arr[], int n);
struct debts
{
    char name[50];
    double amount;
};
int main()
```

```
{
    using namespace std;
    int things[6] = {13, 31, 103, 301, 310, 130};
    struct debts mr_E[3] = {
    {"Ima Wolfe", 2400.0},
    {"Ura Foxe", 1300.0},
    {"Iby Stout", 1800.0}
    };
    double * pd[3];
    for (int i=0; i<3; i++)
        pd[i] = &mr_E[i].amount;
    cout << "Sum: Mr.E's counts of things: "
        << SumArrray(things, 6) << endl;
    cout << "Sum: Mr.E's debts: "
        << SumArrray(pd, 3) << endl;
    return 0;
}
template <typename T>
T SumArrray(T arr[], int n)
{
    using namespace std;
    T sum = 0;
    cout << "template A\n";
    for (int i = 0; i < n; i++)
        sum += arr[i];
    return sum;
}
template <typename T>
T SumArrray(T * arr[], int n)
{
    using namespace std;
    T sum = 0;
    cout << "template B\n";
    for (int i = 0; i < n; i++)
        sum += *arr[i];
    return sum;
}
```

第 9 章 内存模型和名称空间

// ex9.1

```
//golf.h
```

```
const int Len = 40;
struct golf
{
    char fullname[Len];
    int handicap;
};
void setgolf(golf & g, const char * name, int hc);
int setgolf(golf & g);
void handicap(golf & g, int hc);
void showgolf(const golf & g);

//golf.cpp
#include <iostream>
#include "golf.h"
using namespace std;
int setgolf(golf & g)
{
    cout << "Enter the golfer's name: \n";
    cin.get(g.fullname, Len);
    if (g.fullname[0] == '\0')
        return 0;
    cout << "Enter the handicap for " << g.fullname << endl;
    while (!(cin >> g.handicap))
    {
        cin.clear();
        while (cin.get() != '\n')
            continue;
        cout << "Please enter an integer.\n";
    }
    cin.get();
    return 1;
}
void setgolf(golf & g, const char * name, int hc)
{
    strncpy_s(g.fullname, name, Len);
    g.handicap = hc;
}
void handicap(golf & g, int hc)
{
   g.handicap = hc;
}
void showgolf(const golf & g)
{
    cout << "Golfer: " << g.fullname << "\n";
```

```
    cout << "Handicap: " << g.handicap << "\n\n";
}
//main.cpp
#include <iostream>
#include "golf.h"
const int Men = 5;
int main()
{
    golf golfer[Men];
    int i;
    for (i = 0; i < Men; i++)
    {
        if(setgolf(golfer[i]) == 0)
            break;
    }
    for (int j = 0; j < i; j++)
        showgolf(golfer[j]);
    golf ann;
    setgolf(ann, "Ann Birdfree", 24);
    showgolf(ann);
    handicap(ann, 4);
    showgolf(ann);
    return 0;
}
```

//ex9.2

```
#include <iostream>
#include <string>
using namespace std;
void strcount(const string);
int main()
{
    string input;
    cout << "Enter a line:\n";
    getline(cin, input);
    while (input != "")
    {
        strcount(input);
        cout << "Enter next line (empty line to quit):\n";
        getline(cin, input);
    }
```

```
    cout << "Bye\n";
    return 0;
}
void strcount(const string str)
{
    static int total = 0;
    int count;
    count = str.size();
    total += count;
    cout << count << " characters\n";
    cout << total << " characters total\n";
}
//ex9.3
#include <iostream>
#include <new>
#include <cstring>
using namespace std;
struct chaff
{
    char dross[20];
    int slag;
};
int main()
{
    chaff * p = new chaff[2];
    strcpy_s(p[0].dross, "Piffa like");
    p[0].slag = 5;
    strcpy_s(p[1].dross, "Fuck me so hard");
    p[1].slag = 6;
    for (int i = 0; i < 2; i++)
        cout << p[i].dross << " " << p[i].slag << endl;
    return 0;
}
//ex9.4
//sales.h
namespace SALES
{
const int QUARTERS = 4;
struct Sales
{
double sales[QUARTERS];
double average;
double max;
double min;
```

```
};
// copies the lesser of 4 or n items from the array ar
// to the sales member of s and computes and stores the
// average, maximum, and minimum values of the entered items;
// remaining elements of sales, if any, set to 0
void setSales(Sales & s, const double ar[], int n);
// gathers sales for 4 quarters interactively, stores them
// in the sales member of s and computes and stores the
// average, maximum, and minimum values
void setSales(Sales & s);
// display all information in structure s
void showSales(const Sales & s);
}
//Sales.cpp
#include <iostream>
#include "Sales.h"
namespace SALES
{
using std::cout;
using std::cin;
using std::endl;
static double calaverage(double arr[], unsigned arrSize)
{
double sum = 0;
for (int i=0; i<arrSize; i++)
sum += arr[i];
return sum/arrSize;
}
static double calmax(double arr[], unsigned arrSize)
{
double max = arr[0];
for (int i=1; i<arrSize; i++)
{
if (max < arr[i])
max = arr[i];
}
return max;
}
static double calmin(double arr[], unsigned arrSize)
{
double min = arr[0];
for (int i=1; i<arrSize; i++)
{
if (min > arr[i])
```

```
min = arr[i];
}
return min;
}
void setSales(Sales & s, const double ar[], int n)
{
unsigned times = n < QUARTERS ? (unsigned)n : QUARTERS;
for (int i=0; i<times; i++)
s.sales[i] = ar[i];
for (int i=times; i<QUARTERS; i++)
s.sales[i] = 0;
s.average = calaverage(s.sales, times);
s.max = calmax(s.sales, times);
s.min = calmin(s.sales, times);
}
void setSales(Sales & s)
{
cout << "Enter 4 sales:\n";
for (int i=0; i<QUARTERS; i++)
{
cout << "sales " << i+1 << ": ";
cin >> s.sales[i];
}
s.average = calaverage(s.sales, QUARTERS);
s.max = calmax(s.sales, QUARTERS);
s.min = calmin(s.sales, QUARTERS);
}
void showSales(const Sales & s)
{
cout << "sales: ";
for (int i=0; i<QUARTERS; i++)
cout << s.sales[i] << " ";
cout << endl;
cout << "average: " << s.average << endl;
cout << "max: " << s.max << endl;
cout << "min: " << s.min << endl;
}
}
//main
#include <iostream>
#include "Sales.h"
using namespace std;
int main ()
{
```

```
using namespace SALES;
Sales salesBook;
double salesList[] = {12.2, 11.16, 10.61, 16.24, 11.53};
setSales(salesBook, salesList,
sizeof(salesList)/sizeof(salesList[0]));
showSales(salesBook);
Sales salesPen;
setSales(salesPen);
showSales(salesPen);
}
#include <iostream>
#include "Sale.h"
using namespace std;
int main()
{
    using namespace SALES;
    Sales A, B;
    double h[4] = {1.1, 2.2, 3.3, 4.4};
    setSales(A);
    showSales(A);
    setSales(B);
    showSales(B);
    return 0;
}
```

第 10 章 对象和类

//ex10.1

```
//bankaccount.h
#ifndef BANKACCOUNT_H_
#define BANKACCOUNT_H_
#include <string>
class BankAccount
{
private:
std::string name;
std::string acctnum;
double balance;
public:
BankAccount(const std::string & client,
```

```
const std::string & num, double bal=0.0);
void show() const;
void deposit(double cash);
void withdraw(double cash);
};
#endif
//bankaccount.cpp
#include <iostream>
#include "bankaccount.h"
BankAccount::BankAccount(const std::string & client,
const std::string & num, double bal)
{
name = client;
acctnum = num;
balance = bal;
}
void BankAccount::show()const
{
using std::cout;
using std::endl;
cout << "Client: " << name << endl;
cout << "Account Number: " << acctnum << endl;
cout << "Balance: " << balance << endl;
}
void BankAccount::deposit(double cash)
{
if (cash >= 0)
balance += cash;
else
std::cout << "Illegal transaction attempted";
}
void BankAccount::withdraw(double cash)
{
if (cash < 0)
std::cout << "Illegal transaction attempted";
else if (cash <= balance)
balance -=cash;
else
std::cout << "Request denied due to insufficient funds.\n";
}
//main.cpp
#include <iostream>
#include "bankaccount.h"
int main()
```

```
{
BankAccount ba("Kermit", "croak322", 123.00);
ba.show();
ba.deposit(20);
ba.show();
ba.withdraw(300);
ba.show();
ba.withdraw(23);
ba.show();
return 0;
}
//ex10.2
#ifndef PERSON_H_
#define PERSON_H_
#include <string>
class Person
{
private:
static const int LIMIT=25;
std::string lname;
char fname[LIMIT];
public:
Person() {lname=""; fname[0]='\0';} //#1
Person(const std::string &ln, const char * fn="Heyyou"); //#2
// the following methods display lname and fname
void Show() const; // firstname lastname format
void FormalShow() const; // lastname, firstname format
};
#endif
//person.cpp
#include <iostream>
#include <cstring>
#include "person.h"
Person::Person(const std::string &ln, const char * fn)
{
lname = ln;
strcpy(fname, fn);
}
void Person::Show() const
{
using std::cout;
using std::endl;
cout << "The people's name is " << fname << " "<< lname << endl;
}
```

```
void Person::FormalShow() const
{
using std::cout;
using std::endl;
cout << "The people's name is " << lname << ", "<< fname <<endl;
}
//main.cpp
#include <iostream>
#include "person.h"
int main()
{
using std::cout;
using std::endl;
Person one;
Person two("Smythecraft");
Person three("Dimwiddy", "Sam");
one.Show();
one.FormalShow();
cout << endl;
two.Show();
two.FormalShow();
cout << endl;
three.Show();
three.FormalShow();
cout << endl;
return 0;
}
//ex10.3
//golf.h
#ifndef GOLF_H_
#define GOLF_H_
class Golf
{
private:
static const int Len = 40;
char fullname[Len];
int handicap;
public:
Golf();
Golf(const char * name, int hc);
const Golf & setgolf(const Golf & g);
void showgolf() const;
};
#endif
```

```cpp
//golf.cpp
#include <iostream>
#include <cstring>
#include "golf.h"
Golf::Golf()
{
strcpy(fullname, "No Name");
handicap = 0;
}
Golf::Golf(const char * name, int hc)
{
strcpy(fullname, name);
handicap = hc;
}
const Golf & Golf::setgolf(const Golf & g)
{
strcpy(fullname, g.fullname);
handicap = g.handicap;
return *this;
}
void Golf::showgolf() const
{
std::cout << "Golfer: " << fullname << "\n";
std::cout << "Handicap: " << handicap << "\n\n";
}
//main
#include <iostream>
#include "golf.h"
int main()
{
Golf golger1("Ann Birdfree", 5);
golger1.showgolf();
Golf golger2;
golger2.setgolf(golger1);
golger2.showgolf();
return 0;
}
//ex10.4
//sale.h
#ifndef SALE_H_
#define SALE_H_
namespace SALES
{
const int QUARTERS = 4;
```

```
class Sales
{
private:
double sales[QUARTERS];
double average;
double max;
double min;
public:
// default constructor
Sales();
// copies the lesser of 4 or n items from the array ar
// to the sales member of s and computes and stores the
// average, maximum, and minimum values of the entered items;
// remaining elements of sales, if any, set to 0
Sales(const double ar[], int n);
// gathers sales for 4 quarters interactively, stores them
// in the sales member of s and computes and stores the
// average, maximum, and minimum values
void setSales();
// display all information in structure s
void showSales() const;
};
}
#endif
//sale.cpp
#include <iostream>
#include "sale.h"
namespace SALES
{
using std::cout;
using std::cin;
using std::endl;
static double calaverage(double arr[], unsigned arrSize)
{
double sum = 0;
for (int i=0; i<arrSize; i++)
sum += arr[i];
return sum/arrSize;
}
static double calmax(double arr[], unsigned arrSize)
{
double max = arr[0];
for (int i=1; i<arrSize; i++)
{
```

```
if (max < arr[i])
max = arr[i];
}
return max;
}
static double calmin(double arr[], unsigned arrSize)
{
double min = arr[0];
for (int i=1; i<arrSize; i++)
{
if (min > arr[i])
min = arr[i];
}
return min;
}
Sales::Sales()
{
min = 0;
max = 0;
average = 0;
for (int i = 0; i < QUARTERS; i++)
sales[i] =0;
}
Sales::Sales(const double ar[], int n)
{
unsigned times = n < QUARTERS ? (unsigned)n : QUARTERS;
for (int i=0; i<times; i++)
sales[i] = ar[i];
for (int i=times; i<QUARTERS; i++)
sales[i] = 0;
average = calaverage(sales, times);
max = calmax(sales, times);
min = calmin(sales, times);
}
void Sales::setSales()
{
cout << "Enter 4 sales:\n";
for (int i=0; i<QUARTERS; i++)
{
cout << "sales " << i+1 << ": ";
cin >> sales[i];
} *
this = Sales(sales, QUARTERS);
}
```

```
void Sales::showSales() const
{
cout << "sales: ";
for (int i=0; i<QUARTERS; i++)
cout << sales[i] << " ";
cout << endl;
cout << "average: " << average << endl;
cout << "max: " << max << endl;
cout << "min: " << min << endl;
}
}
//main.cpp
#include <iostream>
#include "sale.h"
using namespace std;
int main ()
{
using namespace SALES;
double salesList[] = {12.2, 11.16, 10.61, 16.24, 11.53};
Sales salesBook(salesList,
sizeof(salesList)/sizeof(salesList[0]));
salesBook.showSales();
Sales salesPen;
salesPen.setSales();
salesPen.showSales();
return 0;
}
//ex10.5
//stack.h
#ifndef STACK_H_
#define STACK_H_
struct customer{
char fullname[35];
double payment;
};
typedef customer Item;
class Stack
{
private:
enum {MAX = 10};
Item items[MAX];
int top;
public:
Stack();
```

```
bool isempty() const;
bool isfull() const;
// push() returns false if stack already is full, true otherwise
bool push(const Item & item); // add item to stack
// pop() returns false if stack already is empty, true otherwise
bool pop(Item & item); // pop top into item
};
#endif
//stack.cpp
#include <iostream>
#include "stack.h"
Stack::Stack()
{
top = 0;
}
bool Stack::isempty() const
{
return top == 0;
}
bool Stack::isfull() const
{
return top == MAX;
}
bool Stack::push(const Item & item)
{
if (top < MAX)
{
items[top++] = item;
return true;
}
else
return false;
}
bool Stack::pop(Item & item)
{
if (top > 0)
{
item = items[--top];
return true;
}
else
return false;
}
//main.cpp
```

```
#include <iostream>
#include <cctype>
#include "stack.h"
void get_customer(customer & cu);
int main()
{
using namespace std;
Stack st;
customer temp;
double payment = 0;
char ch;
cout << "Please enter A to add a customer,\n"
<< "P to process a customer, and Q to quit.\n";
while (cin >> ch && (ch = toupper(ch)) != 'Q')
{
while (cin.get() != '\n')
continue;
if (ch != 'A' && ch != 'P')
{
cout << "Please respond A, P or Q: ";
continue;
}
switch (ch)
{
case 'A':if (st.isfull())
cout << "stack already full\n";
else
{
get_customer(temp);
st.push(temp);
}
break;
case 'P':if (st.isempty())
cout << "stack is empty\n";
else
{
st.pop(temp);
payment += temp.payment;
cout << temp.fullname << " processed. ";
cout << "Payments now total $"
<< payment << "\n";
}
break;
}
```

```
cout << "Please enter A to add a customer,\n"
<< "P to process a customer, and Q to quit.\n";
}
cout << "Done!\n";
return 0;
}
void get_customer(customer & cu)
{
using namespace std;
cout << "Enter customer name: ";
cin.getline(cu.fullname, 35);
cout << "Enter customer payment: ";
cin >> cu.payment;
while (cin.get() != '\n')
continue;
}
//ex10.6
//move.h
#ifndef MOVE_H_
#define MOVE_H_
class Move
{
private:
double x;
double y;
public:
Move(double a = 0, double b = 0);
void showMove() const;
Move add(const Move & m) const;
//this function adds x of m to x of invoking object to get new x
//add y of m to y of invoking object to get new y, creates a new
//move object initialized to new x,y values and returns it
void reset(double a = 0,double b = 0);
};
#endif
//move.cpp
#include <iostream>
#include "move.h"
Move::Move(double a, double b)
{
x = a;
y = b;
}
void Move::showMove() const
```

```
{
std::cout << "x = " << x
<< ", y = " << y << "\n";
}
Move Move::add(const Move & m) const
{
Move temp;
temp.x = x + m.x;
temp.y = y + m.y;
return temp;
}
void Move::reset(double a, double b)
{
x = a;
y = b;
}
//main
#include <iostream>
#include "move.h"
int main()
{
using std::cout;
using std::endl;
Move move1(4,5);
Move move2(2,1);
Move move3;
cout << "The number in move1 is:\n";
move1.showMove();
cout << "The number in move2 is:\n";
move2.showMove();
move3 = move2.add(move1);
cout << "The number in move3 is :\n";
move3.showMove();
cout << "move1+move2, now move2's number is :\n";
move2.showMove();
cout << "After move1 + move2,now move1's number is :\n";
move1.showMove();
move1.reset();
cout << "After reset move1,now move1's number is:\n";
move1.showMove();
return 0;
}
//ex10.7
//plorg.cpp
```

```
#ifndef PLORG_H_
#define PLORG_H_
class Plorg
{
private:
char name[20];
int CI;
public:
Plorg();
Plorg(char * na, int n = 50);
void resetCI(int n);
void showplorg() const;
};
#endif
//plorg.cpp
#include <iostream>
#include <cstring>
#include "plorg.h"
Plorg::Plorg()
{
strcpy(name, "Plorga");
CI = 0;
}
Plorg::Plorg(char * na, int n)
{
strcpy(name, na);
CI = n;
}
void Plorg::resetCI(int n)
{
CI = n;
}
void Plorg::showplorg() const
{
std::cout << "The plorg's name is " << name << "\n"
<<"The CI is "<< CI <<std::endl;
}
//main.cpp
#include <iostream>
#include "plorg.h"
int main()
{
using namespace std;
Plorg plorg1;
```

```
plorg1.showplorg();
Plorg plorg2("heyyroup", 31);
plorg2.showplorg();
plorg1.resetCI(41);
plorg1.showplorg();
return 0;
}
//ex10.8
//list.h
#ifndef LIST_H_
#define LIST_H_
const int TSIZE = 50;
struct film
{
char title[TSIZE];
int rating;
};
typedef struct film Item;
const int LISTMAX = 10;
class List
{
private:
Item items[LISTMAX];
int count;
public:
List();
bool isempty();
bool isfull();
int itemcount();
bool additem(Item item);
void visit(void (*pf)(Item &));
};
#endif
//list.cpp
#include "list.h"
List::List()
{
count = 0;
}
bool List::isempty()
{
return count == 0;
}
bool List::isfull()
```

```
{
return count == LISTMAX;
}
int List::itemcount()
{
return count;
}
bool List::additem(Item item)
{
if (count == LISTMAX)
return false;
else
items[count++] = item;
return true;
}
void List::visit(void (*pf)(Item &))
{
for (int i=0; i<count; i++)
(*pf)(items[i]);
}
//main.cpp
#include <iostream>
#include <cstdlib>
#include "list.h"
void showfilm(Item & item);
int main()
{
using namespace std;
List movies;
Item temp;
if (movies.isfull())
{
cout << "No more room in list! Bye!\n";
exit(1);
}
cout << "Enter first movie title:\n";
while (cin.getline(temp.title, TSIZE) && temp.title[0] != '\0')
{
cout << "Enter your rating <1-10>: ";
cin >> temp.rating;
while (cin.get() != '\n')
continue;
if (movies.additem(temp) == false)
{
```

```
cout << "List already is full!\n";
break;
}
if (movies.isfull())
{
cout << "You have filled the list.\n";
break;
}
cout << "Enter next movie title (empty line to stop):\n";
}
if (movies.isempty())
cout << "No data entered.";
else
{
cout << "Here is the movie list:\n";
movies.visit(showfilm);
}
cout << "Bye!\n";
return 0;
}
void showfilm(Item & item)
{
std::cout << "Movie: " << item.title << " Rating: "
<< item.rating << std::endl;
}
//11
```

1、

```
//vector.h
#ifndef VECTOR_H_
#define VECTOR_H_

#include <iostream>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <fstream>
using namespace std;

namespace VECTOR
{
    class Vector
    {
    public:
        enum Mode { RECT, POL };
```

```
    private:
        double x;
        double y;
        double mag;
        doubleang;
        Mode mode;

        voidset_mag();
        voidset_ang();
        voidset_x();
        voidset_y();
    public:
        Vector();
        Vector(double n1, double n2, Mode form = RECT);
        void reset(double n1, double n2, Mode form = RECT);
        ~Vector();
        doublexval()const{ return x; }
        doubleyval()const{ return y; }
        doublemagval()const { return mag; }
        doubleangval()const { return ang; }
        voidpolar_mode();
        voidrect_mode();

        Vector operator+(const Vector &b)const;
        Vector operator-(const Vector &b)const;
        Vector operator-()const;
        Vector operator*(double n)const;

        friend Vector operator*(double n, const Vector &a);
        friendostream& operator<<(ostream&os, const Vector &v);
    };
}

#endif

//vector.cpp
#include "vector.h"

namespace VECTOR
{
    const double Rad_to_deg = 45.0 / atan(1.0);

    void Vector::set_mag()
    {
```

```
    mag = sqrt(x * x + y * y);
}

void Vector::set_ang()
{
    if (x == 0.0 && y == 0.0)
        ang = 0.0;
    else
        ang = atan2(y, x);
}

void Vector::set_x()
{
    x = mag * cos(ang);
}

void Vector::set_y()
{
    y = mag * sin(ang);
}

Vector::Vector()
{
    x = y = mag = ang = 0.0;
    mode = RECT;
}

Vector::Vector(double n1, double n2, Mode form)
{
    mode = form;
    if (form == RECT)
    {
        x = n1;
        y = n2;
        set_mag();
        set_ang();
    }
    else if (form == POL)
    {
        mag = n1;
        ang = n2 / Rad_to_deg;
        set_x();
        set_y();
    }
```

```
    else
    {
        cout<< "Incorrect 3rd argument to Vector() -- ";
        cout<< "vector set to 0\n";
        x = y = mag = ang = 0.0;
        mode = RECT;
    }
}

void Vector::reset(double n1, double n2, Mode form)
{
    mode = form;
    if (form == RECT)
    {
        x = n1;
        y = n2;
        set_mag();
        set_ang();
    }
    else if (form == POL)
    {
        mag = n1;
        ang = n2 / Rad_to_deg;
        set_x();
        set_y();
    }
    else
    {
        cout<< "Incorrect 3rd argument to Vector() -- ";
        cout<< "vector set to 0\n";
        x = y = mag = ang = 0.0;
        mode = RECT;
    }
}

Vector::~Vector()
{

}

void Vector::polar_mode()
{
    mode = POL;
}
```

```
void Vector::rect_mode()
{
	mode = RECT;
}

Vector Vector::operator+(const Vector &b)const
{
	return Vector(x + b.x, y + b.y);
}

Vector Vector::operator-(const Vector &b)const
{
	return Vector(x - b.x, y - b.y);
}

Vector Vector::operator-()const
{
	return Vector(-x, -y);
}

Vector Vector::operator*(double n)const
{
	return Vector(n*x, n*y);
}

Vector operator*(double n, const Vector &a)
{
	return a*n;
}

ostream&operator<<(ostream&os, const Vector &v)
{
	if (v.mode == Vector::RECT)
		os<< "(x,y) = (" <<v.x<< ", " <<v.y<< ")";
	else if (v.mode == Vector::POL)
	{
		os<< "(m,a) = (" <<v.mag<< ", "
			<<v.ang*Rad_to_deg<< ")";
	}
	else
		os<< "Vector object mode is invalid";
	returnos;
}
```

```
}

//randwalk.cpp
#include "vector.h"
int main()
{
	using VECTOR::Vector;
	srand(time(0));
	double direction;
	Vector step;
	Vector result(0.0, 0.0);
	unsigned long steps = 0;
	double target;
	doubledstep;
	ofstreamfout;
	fout.open("savesteps.txt");
	cout<< "Enter target distance (q to quit): ";
	while (cin>> target)
	{
		cout<< "Enter step length: ";
		if (!(cin>>dstep))
			break;
		fout<< "Target Distance: " << target << " Step Size: " <<dstep<<endl;
		while (result.magval() < target)
		{
			fout<< steps << ": " << result <<endl;
			direction = rand() % 360;
			step.reset(dstep, direction, Vector::POL);
			result = result + step;
			steps++;
		}
		cout<< "After " << steps << " steps, the subject "
			"has the following location:\n";
		cout<< result <<endl;

		fout<< "After " << steps << " steps, the subject "
			"has the following location:\n";
		fout<< result <<endl;

		result.polar_mode();
		cout<< " or\n" << result <<endl;
		cout<< "Average outward distance per step = "
			<<result.magval() / steps <<endl;
```

```
            fout<< " or\n" << result <<endl;
            fout<< "Average outward distance per step = "
                  <<result.magval() / steps <<endl;
            steps = 0;
            result.reset(0.0, 0.0);
            cout<< "Enter target distance (q to quit): ";
      }
      cout<< "Bye!\n";
      cin.clear();
      while (cin.get() != '\n')
            continue;
      cin.get();
      return 0;
}
```

2、

```
//vector.h
#ifndef VECTOR_H_
#define VECTOR_H_

#include <iostream>
#include <cmath>
#include <cstdlib>
#include <ctime>

using namespace std;

namespace VECTOR
{
      class Vector
      {
      public:
            enum Mode { RECT, POL };
      private:
            double x;
            double y;
            Mode mode;

            doubleset_mag()const;
            doubleset_ang()const;
            voidset_x(double mag, double ang);
            voidset_y(double mag, double ang);
      public:
            Vector();
```

```
        Vector(double n1, double n2, Mode form = RECT);
        void reset(double n1, double n2, Mode form = RECT);
        ~Vector();
        doublexval()const{ return x; }
        doubleyval()const{ return y; }
        doublemagval()const { return set_mag(); }
        doubleangval()const { return set_ang(); }
        voidpolar_mode();
        voidrect_mode();

        Vector operator+(const Vector &b)const;
        Vector operator-(const Vector &b)const;
        Vector operator-()const;
        Vector operator*(double n)const;

        friend Vector operator*(double n, const Vector &a);
        friendostream& operator<<(ostream&os, const Vector &v);
    };
}

#endif

//vector.cpp
#include "vector.h"

namespace VECTOR
{
    const double Rad_to_deg = 45.0 / atan(1.0);

    double Vector::set_mag()const
    {
        returnsqrt(x * x + y * y);
    }

    double Vector::set_ang()const
    {
        if (x == 0.0 && y == 0.0)
            return 0.0;
        else
            return atan2(y, x);
    }

    void Vector::set_x(double mag, double ang)
    {
```

```
    x = mag * cos(ang);
}

void Vector::set_y(double mag, double ang)
{
    y = mag * sin(ang);
}

Vector::Vector()
{
    x = y = 0.0;
    mode = RECT;
}

Vector::Vector(double n1, double n2, Mode form)
{
    mode = form;
    if (form == RECT)
    {
        x = n1;
        y = n2;
    }
    else if (form == POL)
    {
        set_x(n1, n2 / Rad_to_deg);
        set_y(n1, n2 / Rad_to_deg);
    }
    else
    {
        cout<< "Incorrect 3rd argument to Vector() -- ";
        cout<< "vector set to 0\n";
        x = y = 0.0;
        mode = RECT;
    }
}

void Vector::reset(double n1, double n2, Mode form)
{
    mode = form;
    if (form == RECT)
    {
        x = n1;
        y = n2;
    }
```

```
    else if (form == POL)
    {
        set_x(n1, n2 / Rad_to_deg);
        set_y(n1, n2 / Rad_to_deg);
    }
    else
    {
        cout<< "Incorrect 3rd argument to Vector() -- ";
        cout<< "vector set to 0\n";
        x = y = 0.0;
        mode = RECT;
    }
}

Vector::~Vector()
{

}

void Vector::polar_mode()
{
    mode = POL;
}

void Vector::rect_mode()
{
    mode = RECT;
}

Vector Vector::operator+(const Vector &b)const
{
    return Vector(x + b.x, y + b.y);
}

Vector Vector::operator-(const Vector &b)const
{
    return Vector(x - b.x, y - b.y);
}

Vector Vector::operator-()const
{
    return Vector(-x, -y);
}
```

```
    Vector Vector::operator*(double n)const
    {
        return Vector(n*x, n*y);
    }

    Vector operator*(double n, const Vector &a)
    {
        return a*n;
    }

    ostream&operator<<(ostream&os, const Vector &v)
    {
        if (v.mode == Vector::RECT)
            os<< "(x,y) = (" <<v.x<< ", " <<v.y<< ")";
        else if (v.mode == Vector::POL)
        {
            os<< "(m,a) = (" <<v.set_mag() << ", "
                <<v.set_ang()*Rad_to_deg<< ")";
        }
        else
            os<< "Vector object mode is invalid";
        returnos;
    }
}

//randwalk.cpp
#include "vector.h"

int main()
{
    using VECTOR::Vector;
    srand(time(0));
    double direction;
    Vector step;
    Vector result(0.0, 0.0);
    unsigned long steps = 0;
    double target;
    doubledstep;
    cout<< "Enter target distance (q to quit): ";
    while (cin>> target)
    {
        cout<< "Enter step length: ";
        if (!(cin>>dstep))
            break;
```

```
            while (result.magval() < target)
            {
                direction = rand() % 360;
                step.reset(dstep, direction, Vector::POL);
                result = result + step;
                steps++;
            }
            cout<< "After " << steps << " steps, the subject "
                "has the following location:\n";
            cout<< result <<endl;
            result.polar_mode();
            cout<< " or\n" << result <<endl;
            cout<< "Average outward distance per step = "
                <<result.magval() / steps <<endl;
            steps = 0;
            result.reset(0.0, 0.0);
            cout<< "Enter target distance (q to quit): ";
        }
        cout<< "Bye!\n";
        cin.clear();
        while (cin.get() != '\n')
            continue;
        cin.get();
        return 0;
}
```

3、

```
//vector.h
#ifndef VECTOR_H_
#define VECTOR_H_

#include <iostream>
#include <cmath>
#include <cstdlib>
#include <ctime>

using namespace std;

namespace VECTOR
{
        class Vector
        {
        public:
            enum Mode { RECT, POL };
```

```
    private:
        double x;
        double y;
        double mag;
        doubleang;
        Mode mode;

        voidset_mag();
        voidset_ang();
        voidset_x();
        voidset_y();
    public:
        Vector();
        Vector(double n1, double n2, Mode form = RECT);
        void reset(double n1, double n2, Mode form = RECT);
        ~Vector();
        doublexval()const{ return x; }
        doubleyval()const{ return y; }
        doublemagval()const { return mag; }
        doubleangval()const { return ang; }
        voidpolar_mode();
        voidrect_mode();

        Vector operator+(const Vector &b)const;
        Vector operator-(const Vector &b)const;
        Vector operator-()const;
        Vector operator*(double n)const;

        friend Vector operator*(double n, const Vector &a);
        friendostream& operator<<(ostream&os, const Vector &v);
    };
}

#endif

//vector.cpp
#include "vector.h"

namespace VECTOR
{
    const double Rad_to_deg = 45.0 / atan(1.0);

    void Vector::set_mag()
    {
```

```
    mag = sqrt(x * x + y * y);
}

void Vector::set_ang()
{
    if (x == 0.0 && y == 0.0)
        ang = 0.0;
    else
        ang = atan2(y, x);
}

void Vector::set_x()
{
    x = mag * cos(ang);
}

void Vector::set_y()
{
    y = mag * sin(ang);
}

Vector::Vector()
{
    x = y = mag = ang = 0.0;
    mode = RECT;
}

Vector::Vector(double n1, double n2, Mode form)
{
    mode = form;
    if (form == RECT)
    {
        x = n1;
        y = n2;
        set_mag();
        set_ang();
    }
    else if (form == POL)
    {
        mag = n1;
        ang = n2 / Rad_to_deg;
        set_x();
        set_y();
    }
```

```
    else
    {
        cout<< "Incorrect 3rd argument to Vector() -- ";
        cout<< "vector set to 0\n";
        x = y = mag = ang = 0.0;
        mode = RECT;
    }
}

void Vector::reset(double n1, double n2, Mode form)
{
    mode = form;
    if (form == RECT)
    {
        x = n1;
        y = n2;
        set_mag();
        set_ang();
    }
    else if (form == POL)
    {
        mag = n1;
        ang = n2 / Rad_to_deg;
        set_x();
        set_y();
    }
    else
    {
        cout<< "Incorrect 3rd argument to Vector() -- ";
        cout<< "vector set to 0\n";
        x = y = mag = ang = 0.0;
        mode = RECT;
    }
}

Vector::~Vector()
{

}

void Vector::polar_mode()
{
    mode = POL;
}
```

```
void Vector::rect_mode()
{
	mode = RECT;
}

Vector Vector::operator+(const Vector &b)const
{
	return Vector(x + b.x, y + b.y);
}

Vector Vector::operator-(const Vector &b)const
{
	return Vector(x - b.x, y - b.y);
}

Vector Vector::operator-()const
{
	return Vector(-x, -y);
}

Vector Vector::operator*(double n)const
{
	return Vector(n*x, n*y);
}

Vector operator*(double n, const Vector &a)
{
	return a*n;
}

ostream&operator<<(ostream&os, const Vector &v)
{
	if (v.mode == Vector::RECT)
		os<< "(x,y) = (" <<v.x<< ", " <<v.y<< ")";
	else if (v.mode == Vector::POL)
	{
		os<< "(m,a) = (" <<v.mag<< ", "
			<<v.ang*Rad_to_deg<< ")";
	}
	else
		os<< "Vector object mode is invalid";
	returnos;
}
```

```
}

//randwalk.cpp
#include "vector.h"

int main()
{
    using VECTOR::Vector;
    srand(time(0));
    double direction;
    Vector step;
    Vector result(0.0, 0.0);
    unsigned long steps = 0;
    double target;
    doubledstep;
    doublenumbers,N;
    double Min, Max, Sum, Average;
    cout<< "Enter target distance: ";
    cin>> target;
    cout<< "Enter step length: ";
    cin>>dstep;
    cout<< "Enter test numbers: ";
    cin>> numbers;
    N = numbers;
    Min = Max = Sum = Average = 0.0;
    while (numbers)
    {
        while (result.magval() < target)
        {
            direction = rand() % 360;
            step.reset(dstep, direction, Vector::POL);
            result = result + step;
            steps++;
        }
        cout<< "After " << steps << " steps once a walk\n";
        if (Min == 0 || Max == 0)
            Min = Max = steps;
        if (Min > steps)
            Min = steps;
        if (Max < steps)
            Max = steps;
        Sum += steps;
        numbers--;
        steps = 0;
```

```
            result.reset(0.0, 0.0);
        }
        Average = Sum / N;
        cout<< "Max steps is " << Max <<endl;
        cout<< "Min steps is " << Min <<endl;
        cout<< "Average steps is " << Average <<endl;
        cout<< "Bye!\n";
        cin.clear();
        while (cin.get() != '\n')
            continue;
        cin.get();
        return 0;
}
```

4、

```
//mytime.h
#ifndef MYTIME_H_
#define MYTIME_H_

#include <iostream>
#include <string>
#include <stdio.h>

using namespace std;

class Time
{
private:
        int hours;
        int minutes;
public:
        Time();
        Time(int h, int m = 0);
        voidAddMin(int m);
        voidAddHr(int h);
        void Reset(int h = 0, int m = 0);

        Time operator*(double n)const;
        friend Time operator-(const Time &t1, const Time &t2);
        friend Time operator+(const Time &t1, const Time &t2);
        friend Time operator*(double m, const Time &t)
        {
            return t * m;
        }
```

```
        friendostream& operator<<(ostream&os, const Time &t);
};

#endif

//mytime.cpp
#include "mytime.h"

Time::Time()
{
        hours = minutes = 0;
}

Time::Time(int h, int m)
{
        hours = h;
        minutes = m;
}

void Time::AddMin(int m)
{
        minutes += m;
        hours += minutes / 60;
        minutes %= 60;
}

void Time::AddHr(int h)
{
        hours += h;
}

void Time::Reset(int h, int m)
{
        hours = h;
        minutes = m;
}

Time operator+(const Time &t1, const Time &t2)
{
        Time sum;
        sum.minutes = t1.minutes + t2.minutes;
        sum.hours = t1.hours + t2.hours + sum.minutes / 60;
        sum.minutes %= 60;
        return sum;
```

```
}

Time operator-(const Time &t1, const Time &t2)
{
	Time diff;
	int tot1, tot2;
	tot1 = t1.minutes + 60 * t1.hours;
	tot2 = t2.minutes + 60 * t2.hours;
	diff.minutes = (tot1 - tot2) % 60;
	diff.hours = (tot1 - tot2) / 60;
	return diff;
}

Time Time::operator*(double mult)const
{
	Time result;
	longtotalminutes = hours * mult * 60 + minutes * mult;
	result.hours = totalminutes / 60;
	result.minutes = totalminutes % 60;
	return result;
}

ostream&operator<<(ostream&os, const Time &t)
{
	os<<t.hours<< " hours, " <<t.minutes<< " minutes";
	returnos;
}

//usetime.cpp
#include "mytime.h"

int main()
{
	Time aida(3, 35);
	Time tosca(2, 48);
	Time temp;

	cout<< "Aida and Tosca:\n";
	cout<<aida<< "; " <<tosca<<endl;
	temp = aida + tosca;
	cout<< "Aida + Tosca: " << temp <<endl;
	temp = aida - tosca;
	cout<< "Aida - Tosca: " << temp <<endl;
	temp = aida * 1.17;
```

```
        cout<< "Aida * 1.17: " << temp <<endl;
        cout<< "10.0 * Tosca: " << 10.0 * tosca<<endl;

        cin.get();
        return 0;
}

5、
//stonewt.h
#ifndef STONEWT_H_
#define STONEWT_H_

#include <iostream>
#include <stdio.h>
#include <string>

using namespace std;

classStonewt
{
public:
        enum Mode { STN, INPD, FPD };
private:
        staticintconstLbs_per_stn = 14;
        int stone;
        doublepds_left;
        double pounds;
        intpounds_int;
        Mode mode;
        voidset_stn();
        voidset_pds();
        voidset_pds_int();
public:
        Stonewt(double lbs, Mode form);
        Stonewt(intstn, double lbs, Mode form);
        Stonewt();
        ~Stonewt();
        voidstn_mode();
        voidpds_mode();
        voidint_pds_mode();
        operatorint()const;
        operator double()const;
        Stonewt operator+(constStonewt&st)const;
        Stonewt operator-(constStonewt&st)const;
```

```
        Stonewt operator*(double n)const;

        friendStonewt operator*(double n, constStonewt&st);
        friendostream&operator<<(ostream&os, constStonewt&st);
};

#endif

//stonewt.cpp
#include "stonewt.h"


voidStonewt::set_stn()
{
        stone = int(pounds) / Lbs_per_stn;
        pds_left = int(pounds) % Lbs_per_stn + pounds - int(pounds);
}

voidStonewt::set_pds()
{
        pounds = stone*Lbs_per_stn + pds_left;
}

voidStonewt::set_pds_int()
{
        pounds_int = int(pounds);
}

Stonewt::Stonewt(double lbs, Mode form)
{
        mode = form;
        if (form == STN)
        {
                stone = int(lbs) / Lbs_per_stn;
                pds_left = int(lbs) % Lbs_per_stn + lbs - int(lbs);
                set_pds();
                set_pds_int();
        }
        else if (form == INPD)
        {
                pounds_int = int(lbs);
                pounds = lbs;
                set_stn();
        }
```

```
    else if (form == FPD)
    {
        pounds = lbs;
        set_pds_int();
        set_stn();
    }
    else
    {
        cout<< "Incorrect 3rd argument to Stonewt() -- ";
        cout<< "Stonewt set to 0\n";
        stone = pounds = pds_left = 0;
        mode = STN;
    }
}

Stonewt::Stonewt(intstn, double lbs, Mode form)
{
    mode = form;

    if (form == STN)
    {
        stone = stn;
        pds_left = lbs;
        set_pds();
        set_pds_int();
    }
    else if (form == INPD)
    {
        pounds_int = int(stn*Lbs_per_stn + lbs);
        pounds = stn*Lbs_per_stn + lbs;
        set_stn();
    }
    else if (form == FPD)
    {
        pounds = stn*Lbs_per_stn + lbs;
        set_pds_int();
        set_stn();
    }
    else
    {
        cout<< "Incorrect 3rd argument to Stonewt() -- ";
        cout<< "Stonewt set to 0\n";
        stone = pounds = pds_left = 0;
        mode = STN;
```

```
        }
}

Stonewt::Stonewt()
{
        stone = pounds = pds_left = 0;
        mode = STN;
}

Stonewt::~Stonewt()
{

}

voidStonewt::stn_mode()
{
        mode = STN;
}

voidStonewt::pds_mode()
{
        mode = FPD;
}

voidStonewt::int_pds_mode()
{
        mode = INPD;
}

Stonewt::operator int()const
{
        returnint(pounds + 0.5);
}

Stonewt::operator double()const
{
        return pounds;
}

StonewtStonewt::operator+(constStonewt&st)const
{
        returnStonewt(pounds + st.pounds, st.mode);
}
```

```
StonewtStonewt::operator-(constStonewt&st)const
{
	returnStonewt(pounds - st.pounds, st.mode);
}

StonewtStonewt::operator*(double n)const
{
	returnStonewt(pounds*n, mode);
}

Stonewt operator*(double n, constStonewt&st)
{
	returnStonewt(n*st.pounds, st.mode);
}

ostream&operator<<(ostream&os, constStonewt&st)
{
	if (st.mode == Stonewt::STN)
		os<<st.stone<< " stone, " <<st.pds_left<< " pounds\n";
	else if (st.mode == Stonewt::INPD)
		os<<st.pounds_int<< " pounds(int)\n";
	else if (st.mode == Stonewt::FPD)
		os<<st.pounds<< " pounds(double)\n";
	else
		os<< "Error in type\n";
	returnos;
}

//stone.cpp
#include "stonewt.h"

int main()
{
	Stonewt   incognito(275,Stonewt::FPD);
	Stonewtwolfe(285.7,Stonewt::STN);
	Stonewttaft(21, 8,Stonewt::INPD);
	Stonewt temp;
	cout<< "The celebrity weighed ";
	cout<< incognito <<endl;
	cout<< "The detective weighed ";
	cout<<wolfe<<endl;
	cout<< "The President weighed ";
	cout<<taft<<endl;
	temp = incognito + wolfe;
```

```
        cout<< "Incognito + Wolfe = " << temp <<endl;
        temp = wolfe - incognito;
        cout<< "Wolfe - Incognito = " << temp <<endl;
        temp = taft * 10.0;
        cout<< "Taft * 10.0 = " << temp <<endl;
        temp = 10.0 * taft;
        cout<< "10.0 * Taft = " << temp <<endl;

        cin.get();
        return 0;
}
```

6、

```
//stonewt.h
#ifndef STONEWT_H_
#define STONEWT_H_

#include <iostream>
#include <stdio.h>
#include <string>

using namespace std;

classStonewt
{
private:
        enum { Lbs_per_stn = 14 };
        int stone;
        doublepds_left;
        double pounds;
public:
        Stonewt(double lbs);
        Stonewt(intstn, double lbs);
        Stonewt();
        ~Stonewt();
        bool operator<(constStonewt&st)const;
        bool operator<=(constStonewt&st)const;
        bool operator>(constStonewt&st)const;
        bool operator>=(constStonewt&st)const;
        bool operator==(constStonewt&st)const;
        bool operator!=(constStonewt&st)const;
        friendostream&operator<<(ostream&os, constStonewt&st);
};
```

```
#endif
//stonewt.cpp
#include "stonewt.h"

Stonewt::Stonewt(double lbs)
{
	stone = int(lbs) / Lbs_per_stn;
	pds_left = int(lbs) % Lbs_per_stn + lbs - int(lbs);
	pounds = lbs;
}

Stonewt::Stonewt(intstn, double lbs)
{
	stone = stn;
	pds_left = lbs;
	pounds = stn * Lbs_per_stn + lbs;
}

Stonewt::Stonewt()
{
	stone = pounds = pds_left = 0;
}

Stonewt::~Stonewt()
{

}

boolStonewt::operator<(constStonewt&st)const
{
	if (pounds <st.pounds)
		return true;
	else
		return false;
}
boolStonewt::operator<=(constStonewt&st)const
{
	if (pounds <= st.pounds)
		return true;
	else
		return false;
}
boolStonewt::operator>(constStonewt&st)const
{
```

```
        if (pounds >st.pounds)
                return true;
        else
                return false;
}
boolStonewt::operator>=(constStonewt&st)const
{
        if (pounds >= st.pounds)
                return true;
        else
                return false;
}
boolStonewt::operator==(constStonewt&st)const
{
        if (pounds == st.pounds)
                return true;
        else
                return false;
}
boolStonewt::operator!=(constStonewt&st)const
{
        if (pounds != st.pounds)
                return true;
        else
                return false;
}

ostream&operator<<(ostream&os, constStonewt&st)
{
        os<<st.pounds<< " pounds\n";
        returnos;
}
//stone.cpp
#include "stonewt.h"

int main()
{
        Stonewt   sw[6] = { 10.0, 11.0, 12.5 };
        Stonewttemp(11.0);
        for (inti = 3; i< 6; i++)
        {
                double input;
                cout<< "Enter #" <<i + 1 << ": ";
                cin>> input;
```

```
            sw[i] = input;
        }
        for (inti = 0; i< 6; i++)
            cout<< "#" <<i<< ": " <<sw[i];
        int count = 0;
        Stonewt Min = sw[0];
        Stonewt Max = sw[0];
        for (inti = 0; i< 6; i++)
        {
            if (Min >sw[i])
                Min = sw[i];
            if (Max <sw[i])
                Max = sw[i];
            if (temp >= sw[i])
                count++;
        }
        cout<< "The Min pounds: " << Min;
        cout<< "The Max pounds: " << Max;
        cout<< "The numbers not under 11 pounds: " << count;
        cin.get();
        cin.get();
        return 0;
}
```

7、

```
//complexh.h
#ifndef COMPLEX_H_
#define COMPLEX_H_

#include <iostream>
#include <string>
#include <stdio.h>
#include <cmath>

using namespace std;

class Complex
{
private:
        double real;
        double imaginary;
public:
        Complex();
        Complex(double n1);
```

```
        Complex(double n1, double n2);
        ~Complex();
        Complex operator+(const Complex &c)const;
        Complex operator-(const Complex &c)const;
        Complex operator*(const Complex &c)const;
        Complex operator*(double n)const;
        Complex operator~()const;

        friend Complex operator*(double n, const Complex &c);
        friendostream&operator<<(ostream&os, const Complex &c);
        friendistream&operator>>(istream&is, Complex &c);
};

#endif
//complex.cpp
#include "complexh.h"

Complex::Complex()
{
        real = 0.0;
        imaginary = 0.0;
}

Complex::Complex(double n1)
{
        real = n1;
        imaginary = 0.0;
}

Complex::Complex(double n1, double n2)
{
        real = n1;
        imaginary = n2;
}

Complex::~Complex()
{

}

Complex Complex::operator+(const Complex &c)const
{
        return Complex(real + c.real, imaginary + c.imaginary);
}
```

```
Complex Complex::operator-(const Complex &c)const
{
	return Complex(real - c.real, imaginary - c.imaginary);
}

Complex Complex::operator*(const Complex &c)const
{
	doublereal_s;
	doubleimaginary_s;
	real_s = real*c.real - imaginary*c.imaginary;
	imaginary_s = real*c.imaginary + imaginary*c.real;
	return Complex(real_s, imaginary_s);
}

Complex Complex::operator*(double n)const
{
	return Complex(n*real, n*imaginary);
}
Complex Complex::operator~()const
{
	return Complex(real, -imaginary);
}

Complex operator*(double n, const Complex &c)
{
	return Complex(n*c.real, n*c.imaginary);
}

ostream&operator<<(ostream&os, const Complex &c)
{
	os<< "(" <<c.real<< ", " <<c.imaginary<<"i)";
	returnos;
}

istream&operator>>(istream&is, Complex &c)
{
	cout<< "Real: ";
	if (is >>c.real)
	{
		cout<< "Imaginary: ";
		is>>c.imaginary;
	}
	return is;
```

```
}
//useComplex.cpp
#include "complexh.h"

int main()
{
	Complex a(3.0, 4.0);
	Complex c;
	charch;
	cout<< "Enter a complex number (q to quit): ";
	while (cin>>ch)
	{
		if (ch == 'q' || ch == 'Q')
			break;
		else
		{
			cin>> c;
			cout<< "c is " << c << '\n';
			cout<< "Complex conjugate is " << ~c << '\n';
			cout<< "a is " << a << '\n';
			cout<< "a + c is " << a + c << '\n';
			cout<< "a - c is " << a - c << '\n';
			cout<< "a * c is " << a * c << '\n';
			cout<< "2 * c is " << 2 * c << '\n';
		}
		cout<< "Enter a complex number (q to quit): ";

	}
	cout<< "Done!\n";
	cin.get();
	cin.get();
	return 0;
}
```

第十二章编程练习答案

12.1 根据以下类声明，完成类，并编小程序使用它

```
//12.1 根据以下类声明，完成类，并编小程序使用它
#include <iostream>
#include <cstring>
using namespace std;

class Cow{
  char name[20];
```

```
  char * hobby;
  double weight;
public:
  Cow();
  Cow(const char * nm, const char * ho, double wt);
  Cow(const Cow & C);
  ~Cow();
  void ShowCow() const;
};

Cow::Cow(){}
Cow::Cow(const char * nm, const char * ho, double wt)
{
  strcpy(name,nm);
  hobby=new char[strlen(ho)+1];
  strcpy(hobby,ho);
  weight=wt;
}
Cow::Cow(const Cow & C)
{
  strcpy(name,C.name);
  hobby=new char[strlen(C.hobby)+1];
  strcpy(hobby,C.hobby);
  weight=C.weight;
}
Cow::~Cow() {delete [] hobby;}
void Cow::ShowCow() const
{
  cout << name << endl;
  cout << hobby << endl;
  cout << weight << endl;
}

int main()
{
  Cow cow;
  Cow ccc("adads","dsdfsad",34);
  cow=ccc;
  cow.ShowCow();
  ccc.ShowCow();
}
```

12.2 根据以下的主函数，编写类，使得：

a.重载+，使得两个字符串可以合并为一个

b.使用 Stringlow()成员函数，使得字母可以转换为小写

c.使用 Stringup()成员函数，使得字母可转换为大写

d.提供一个成员函数，使它返回一个 char 字符出现的个数

```
//12.2 根据以下的主函数，编写类，使得：
//a.重载+，使得两个字符串可以合并为一个
//b.使用 Stringlow()成员函数，使得字母可以转换为小写
//c.使用 Stringup()成员函数，使得字母可转换为大写
//d.提供一个成员函数，使它返回一个 char 字符出现的个数
#include <iostream>
#include <cstring>
#include <cctype>
using namespace std;

class String
{
  char*          mp_text;
  unsigned m_text_length;
  void assignMember (const char* text)
  {
    m_text_length = strlen(text);
    mp_text = new char [m_text_length + 1];
    strcpy(mp_text, text);
  }
public:
  static const unsigned     k_buffer_max_size = 256;
  const char* toCstr () const
  {
    return (mp_text);
  }
  String (const char* text = "")
  {
    assignMember(text);
  }
  String (const String& str)
  {
    assignMember(str.toCstr());
  }
  ~String ()
  {
    delete [] mp_text;
  }
  unsigned getLength () const
  {
    return (m_text_length);
  }
```

```
void stringup ()
{
  for (unsigned i = 0; i < m_text_length; ++i)
    mp_text[i] = (char)toupper(mp_text[i]);
}
void stringlow ()
{
  for (unsigned i = 0; i < m_text_length; ++i)
    mp_text[i] = (char)tolower((int)mp_text[i]);
}
unsigned has (char ch) const
{
  unsigned     cnt = 0;
  for (unsigned i = 0; i < m_text_length; ++i)
    if (ch == mp_text[i])
      ++cnt;
  return (cnt);
}
String& operator= (const String& str)
{
  if (&str == this)
    return (*this);
  delete [] mp_text;
  assignMember(str.toCstr());
  return (*this);
}
String & operator+= (const String& str)
{
  return (*this += str);
}
char& operator[] (unsigned idx)
{
  return (mp_text[idx]);
}
    const char & operator[] (unsigned idx) const
{
  return (mp_text[idx]);
}
friend ostream & operator<< (ostream& os, const String& str)
{
  os << str.toCstr();
  return (os);
}
friend istream & operator>> (istream& is, String& str)
```

```
  {
    char   txt[k_buffer_max_size];
    if (is >> txt)
      str = txt;
    is.ignore(k_buffer_max_size, '\n');
    return (is);
  }
  friend bool operator< (const String& lvalue, const String& rvalue)
  {
    return (strcmp(lvalue.toCstr(), rvalue.toCstr()) < 0);
  }
  friend bool operator> (const String& lvalue, const String& rvalue)
  {
    return (rvalue < lvalue);
  }
  friend bool operator== (const String& lvalue, const String& rvalue)
  {
    return (!(lvalue < rvalue) && !(lvalue > rvalue));
  }
  friend bool operator<= (const String& lvalue, const String& rvalue)
  {
    return (!(lvalue > rvalue));
  }
  friend bool operator>= (const String& lvalue, const String& rvalue)
  {
    return (!(lvalue < rvalue));
  }
  friend String operator+ (const String& lvalue, const String& rvalue)
  {
    char* p_txt = new char [lvalue.getLength() + rvalue.getLength() + 1];
    strcpy(p_txt, lvalue.toCstr());
    strcat(p_txt, rvalue.toCstr());
    String tmp(p_txt);
    delete [] p_txt;
    return (tmp);
  }
};

int main()
{
  String s1(" and I am a C++ student.");
  String s2 = "Please enter your name: ";
  String s3;
  cout << s2;
```

```
  // overloaded << operator
  cin >> s3;
  // overloaded >> operator
  s2 = "My name is " + s3;
  // overloaded =, + operators
  cout << s2 << ".\n";
  s2 = s2 + s1;
  s2.stringup();
  // converts string to uppercase
  cout     << "The string\n" << s2 << "\ncontains " << s2.has('A')
     << " 'A' characters in it.\n";
  s1 = "red";
  // tstring(const char *),
  // then tstring & operator=(const string&)
  String rgb[3] = { String(s1), String("green"), String("blue")};
  cout << "enter the name of a primary color for mixing light: ";
  String ans;
  bool success = false;
  while (cin >> ans)
  {
    ans.stringlow();
    // converts string to lowercase
    for (int i = 0; i < 3; i++)
    {
      if (ans == rgb[i]) // overloaded == operator
      {
        cout << "That's right!\n";
        success = true;
        break;
      }
    }
    if (success)
      break;
    else
      cout << "Try again!\n";
  }
  cout << "Bye" << endl;
}
```

12.3 重新编写程序清单 10.7,10.8，使用动态内存并重载<<代替 show()

```
//12.3 重新编写程序清单 10.7,10.8，使用动态内存并重载<<代替 show()
#include <iostream>
#include <cstring>
using namespace std;
```

```
class Stock{
  char *company;
  int shares;
  double share_val;
  double total_val;
  void set_tot(){total_val=shares*share_val;};
public:
  Stock(){
    company=new char[8];
    strcpy(company,"no name");
    shares=0;
    share_val=0.0;
    total_val=0.0;
  }
  Stock(const char *co,long n=0,double pr=0)
  {
    int len=strlen(co);
    company=new char[len+1];
    strcpy(company,co);
    if(n<0)
    {
      cout<<"Number of shares can't be negative;"
        <<company<<" shares set to 0"<<endl;
      shares=0;
    }
    else
      shares=n;
    share_val=pr;
    set_tot();
  }
  ~Stock()
  {
    delete []company;
  }
  void buy(long num,double price)
  {
    if(num<0)
    {
      cout<<"Number of shares purchase can't be negative."
        <<" Transaction is aborted."<<endl;
    }
    else
    {
      shares+=num;
```

```
    share_val=price;
    set_tot();
  }
}
void sell(long num,double price)
{
  if(num<0)
  {
    cout<<"Number of shares sold can't be negative."
      <<"Transaction  is  aborted."<<endl;
  }
  else if(num>shares)
  {
     cout<<"You can't sell more than you have!"
       <<"Transaction  is  aborted."<<endl;
  }
  else
  {
    shares-=num;
    share_val=price;
    set_tot();
  }
}
void update(double price)
{
  share_val=price;
  set_tot();
}
const Stock &topval(const Stock &s)const
{
  if(s.total_val>total_val)
    return s;
  else
    return *this;
}
friend ostream &operator<<(ostream &os,const Stock &s)
{
  ios_base::fmtflags orig=os.setf(ios_base::fixed,ios_base::floatfield);
  streamsize prec=os.precision(3);
  os<<"Company:"<<s.company
    <<"  Shares:"<<s.shares<<endl;
  os<<" Share Price:$"<<s.share_val;
  os.precision(2);
  os<<"   Total Worth:&"<<s.total_val<<endl;
```

```
    os.setf(orig,ios_base::floatfield);
    os.precision(prec);
    return os;
  }
};

const int STKS = 4;
int main()
{
// create an array of initialized objects
  Stock stocks[STKS] = {
    Stock("NanoSmart", 12, 20.0),
    Stock("Boffo Objects", 200, 2.0),
    Stock("Monolithic Obelisks", 130, 3.25),
    Stock("Fleep Enterprises", 60, 6.5)
    };
  cout << "Stock holdings:\n";
  int st;
  for (st = 0; st < STKS; st++)
    cout << stocks[st];
// set pointer to first element
  const Stock * top = &stocks[0];
  for (st = 1; st < STKS; st++)
    top = &top->topval(stocks[st]);
// now top points to the most valuable holding
  cout << "\nMost valuable holding:\n";
  cout << *top;
  return 0;
}
```

12.4 按以下类声明，完成类，并编写一个演示程序

```
//12.4 按以下类声明，完成类，并编写一个演示程序
#include <iostream>
using namespace std;
typedef unsigned long Item;

class Stack{
  enum{MAX=10};
  Item * items;
  int size;
  int top;
public:
  Stack(int n=MAX)
  {
    items=new Item [MAX];
```

```
  top=0;
  size=0;
}
Stack(const Stack &st)
{
  items=new Item[st.size];
  top=0;
  size=0;
  for(int i=0;i<st.size;i++)
  {
    items[i]=st.items[i];
    size++;
    top++;
  }
}
~Stack()
{
 delete [] items;
}
bool isEmpty()
{
  return top==0;
}
bool isFull()
{
  return top==MAX;
}
bool push(const Item &it)
{
  if(isFull())
    cout<<"error! Stack is full!"<<endl;
  else
  {
    items[top++]=it;
    size++;
    return true;
  }
  return false;
}
bool pop(Item &item)
{
  if(isEmpty())
    cout<<"error! Stack is empty!"<<endl;
  else
```

```
    {
      item=items[top--];
      size--;
      return true;
    }
    return false;
  }
  Stack & operator = (Stack &st)
  {
    delete [] items;
    items=new Item[st.size];
    top=0;
    size=0;
    for(int i=0;i<st.size;i++)
    {
      items[i]=st.items[i];
      size++;
      top++;
    }
    return (*this);
  }
  friend ostream & operator<<(ostream &os,const Stack & st)
  {
    os<<"This Stack is:"<<endl;
    int len=st.top-1;
    while(len!=-1)
    {
      cout<<st.items[len]<<endl;
      len--;
    }
    return os;
  }
};

int main ()
{
  Stack s;
  Item it[20]={0};
  for(int i=0;i<11;i++)
  {
    it[i]=i+1;
    s.push(it[i]);
  }
  cout<<s;
```

```
  Stack s1(s);
  cout<<"s1="<<s1;
  Stack s2=s;
  cout<<s;
}
```

12.5-12.6 银行 ATM 顾客系统

```
// queue.h -- interface for a queue
#ifndef QUEUE_H_
#define QUEUE_H_
// This queue will contain Customer items
class Customer
{
private:
  long arrive;        // arrival time for customer
  int processtime;    // processing time for customer
public:
  Customer() : arrive(0), processtime (0){}
  void set(long when);
  long when() const { return arrive; }
  int ptime() const { return processtime; }
};

typedef Customer Item;

class Queue
{
private:
// class scope definitions
  // Node is a nested structure definition local to this class
  struct Node { Item item; struct Node * next;};
  enum {Q_SIZE = 10};
// private class members
  Node * front;    // pointer to front of Queue
  Node * rear;        // pointer to rear of Queue
  int items;            // current number of items in Queue
  const int qsize;    // maximum number of items in Queue
  // preemptive definitions to prevent public copying
  Queue(const Queue & q) : qsize(0) { }
  Queue & operator=(const Queue & q) { return *this;}
public:
  Queue(int qs = Q_SIZE); // create queue with a qs limit
  ~Queue();
  bool isempty() const;
  bool isfull() const;
```

```
  int queuecount() const;
  bool enqueue(const Item &item); // add item to end
  bool dequeue(Item &item);   // remove item from front
};
#endif
// queue.cpp -- Queue and Customer methods
#include "queue.h"
#include <cstdlib>           // (or stdlib.h) for rand()

// Queue methods
Queue::Queue(int qs) : qsize(qs)
{
  front = rear = NULL;// or nullptr
  items = 0;
}

Queue::~Queue()
{
  Node * temp;
  while (front != NULL)   // while queue is not yet empty
  {
    temp = front;        // save address of front item
    front = front->next;// reset pointer to next item
    delete temp;      // delete former front
  }
}

bool Queue::isempty() const
{
  return items == 0;
}

bool Queue::isfull() const
{
  return items == qsize;
}

int Queue::queuecount() const
{
  return items;
}

// Add item to queue
bool Queue::enqueue(const Item & item)
```

```
{
  if (isfull())
    return false;
  Node * add = new Node;  // create node
// on failure, new throws std::bad_alloc exception
  add->item = item;      // set node pointers
  add->next = NULL;      // or nullptr;
  items++;
  if (front == NULL)    // if queue is empty,
    front = add;       // place item at front
  else
    rear->next = add;   // else place at rear
  rear = add;                // have rear point to new node
  return true;
}

// Place front item into item variable and remove from queue
bool Queue::dequeue(Item & item)
{
  if (front == NULL)
    return false;
  item = front->item;  // set item to first item in queue
  items--;
  Node * temp = front;// save location of first item
  front = front->next;// reset front to next item
  delete temp;              // delete former first item
  if (items == 0)
    rear = NULL;
  return true;
}

// customer method

// when is the time at which the customer arrives
// the arrival time is set to when and the processing
// time set to a random value in the range 1 - 3
void Customer::set(long when)
{
  processtime = std::rand() % 3 + 1;
  arrive = when;
}
// bank.cpp -- using the Queue interface
// compile with queue.cpp
#include <iostream>
```

```
#include <cstdlib> // for rand() and srand()
#include <ctime>   // for time()
#include "queue.h"
const int MIN_PER_HR = 60;

bool newcustomer(double x); // is there a new customer?

int main()
{
  using std::cin;
  using std::cout;
  using std::endl;
  using std::ios_base;
// setting things up
  std::srand(std::time(0));//  random initializing of rand()

  cout << "Case Study: Bank of Heather Automatic Teller\n";
  cout << "Enter maximum size of queue: ";
  int qs;
  cin >> qs;
  Queue line(qs);           // line queue holds up to qs people

  cout << "Enter the number of simulation hours: ";
  int hours;                 //  hours of simulation
  cin >> hours;
  // simulation will run 1 cycle per minute
  long cyclelimit = MIN_PER_HR * hours; // # of cycles

  cout << "Enter the average number of customers per hour: ";
  double perhour;           //  average # of arrival per hour
  cin >> perhour;
  double min_per_cust;//  average time between arrivals
  min_per_cust = MIN_PER_HR / perhour;

  Item temp;                 //  new customer data
  long turnaways = 0;  //  turned away by full queue
  long customers = 0;  //  joined the queue
  long served = 0;          //  served during the simulation
  long sum_line = 0;    //  cumulative line length
  int wait_time = 0;    //  time until autoteller is free
  long line_wait = 0;  //  cumulative time in line


// running the simulation
```

```
  for (int cycle = 0; cycle < cyclelimit; cycle++)
  {
    if (newcustomer(min_per_cust))  // have newcomer
    {
      if (line.isfull())
        turnaways++;
      else
      {
        customers++;
        temp.set(cycle);    // cycle = time of arrival
        line.enqueue(temp); // add newcomer to line
      }
    }
    if (wait_time <= 0 && !line.isempty())
    {
      line.dequeue (temp);    // attend next customer
      wait_time = temp.ptime(); // for wait_time minutes
      line_wait += cycle - temp.when();
      served++;
    }
    if (wait_time > 0)
      wait_time--;
    sum_line += line.queuecount();
  }

// reporting results
  if (customers > 0)
  {
    cout << "customers accepted: " << customers << endl;
    cout << "  customers served: " << served << endl;
    cout << "           turnaways: " << turnaways << endl;
    cout << "average queue size: ";
    cout.precision(2);
    cout.setf(ios_base::fixed, ios_base::floatfield);
    cout << (double) sum_line / cyclelimit << endl;
    cout << " average wait time: "
       << (double) line_wait / served << " minutes\n";
  }
  else
    cout << "No customers!\n";
  cout << "Done!\n";
  // cin.get();
  // cin.get();
  return 0;
```

```
}

//  x = average time, in minutes, between customers
//  return value is true if customer shows up this minute
bool newcustomer(double x)
{
  return (std::rand() * x / RAND_MAX < 1);
}
```

第十三章编程练习答案

13.1 根据 Cd 基类，完成派生出一个 Classic 类，并测试

```
//13.1 根据 Cd 基类，完成派生出一个 Classic 类，并测试
#include <iostream>
#include <cstring>
using namespace std;

// base class
class Cd
{
        char performers[50];
        char label[20];
        int selections; // number of selections
        double playtime; // playing time in minutes
public:
        explicit Cd(const char * s1 = "", const char * s2 = "", int n =
0, double x = 0.0);
        virtual ~Cd() {}
        virtual void Report() const; // reports all CD data
};

static void cpStr (char* p_des_txt, const char* p_src_txt, unsigned
des_arr_size)
{
        unsigned       str_len = strlen(p_src_txt) < des_arr_size-1 ?
strlen(p_src_txt) : des_arr_size-1;
        strncpy(p_des_txt, p_src_txt, str_len);
        p_des_txt[str_len] = '\0';
```

```
}

Cd::Cd (const char * s1, const char * s2, int n, double x)
        : selections(n), playtime(x)
{
        cpStr(performers, s1, 50);
        cpStr(label, s2, 20);
}

void Cd::Report() const
{
        cout << performers << ", " << label << ", " << selections << ",
" << playtime << flush;
}

class Classic : public Cd
{
        static const unsigned     mk_size = 64;
        char   m_songs[mk_size];
public:
        Classic (const char* songs_list = "", const char * s1 = "", const
char * s2 = "", int n = 0, double x = 0.0);
        virtual void Report() const; // reports all CD data
};

Classic::Classic (const char* songs_list, const char * s1, const char *
s2, int n, double x)
        : Cd(s1, s2, n, x)
{
        cpStr(m_songs, songs_list, mk_size);
}

void Classic::Report () const
{
        Cd::Report();
        cout << ", " << m_songs << endl;
}

void Bravo(const Cd & disk)
{
        disk.Report();
        cout << endl;
}
```

```
int main()
{
        Cd c1("Beatles", "Capitol", 14, 35.5);
        Classic c2 = Classic("Piano Sonata in B flat, Fantasia in C",
"Alfred Brendel", "Philips", 2, 57.17);
        Cd *pcd = &c1;
        cout << "Using object directly:\n";
        c1.Report(); // use Cd method
        c2.Report(); // use Classic method
        cout << "Using type cd * pointer to objects:\n";
        pcd->Report(); // use Cd method for cd object
        pcd = &c2;
        pcd->Report(); // use Classic method for classic object
        cout << "Calling a function with a Cd reference argument:\n";
        Bravo(c1);
        Bravo(c2);
        cout << "Testing assignment: ";
        Classic copy;
        copy = c2;
        copy.Report();
}</cstring></iostream>
```

13.2 对 13.1,使用动态内存记录字符串

```
//13.2 对 13.1,使用动态内存记录字符串
#include <iostream>
#include <cstring>
using namespace std;

// base class
class Cd
{
    char* performers;
    char* label;
    int selections; // number of selections
    double playtime; // playing time in minutes
public:
    explicit Cd(const char * s1 = "", const char * s2 = "", int n = 0, double x = 0.0);
    Cd(const Cd & d);
    virtual ~Cd();
    virtual void Report() const; // reports all CD data
    Cd & operator=(const Cd & d);
```

```
};

static char* cpNewStr (const char* p_src_txt)
{
	unsigned	str_len = strlen(p_src_txt);
	char*	p_des_txt = new char [str_len + 1];
	strcpy(p_des_txt, p_src_txt);
	return (p_des_txt);
}

Cd::Cd (const char * s1, const char * s2, int n, double x)
	: selections(n), playtime(x)
{
	performers = cpNewStr(s1);
	label = cpNewStr(s2);
}

Cd::~Cd ()
{
	delete [] performers;
	delete [] label;
}

Cd::Cd(const Cd & d)
	: selections(d.selections), playtime(d.playtime)
{
	performers = cpNewStr(d.performers);
	label = cpNewStr(d.label);
}

Cd & Cd::operator=(const Cd & d)
{
	if (&d == this) {
		return (*this);
	}
	delete [] performers;
	performers = cpNewStr(d.performers);
	delete [] label;
	label = cpNewStr(d.label);
	selections = d.selections;
	playtime = d.playtime;
	return (*this);
}
```

```
void Cd::Report() const
{
	cout << performers << ", " << label << ", " << selections << ", " << playtime << flush;
}

// derive
class Classic : public Cd
{
	char*	songs;
public:
	explicit Classic (const char* songs_list = "", const char * s1 = "", const char * s2 = "", int n = 0, double x = 0.0);
	Classic (const Classic& classic);
	virtual ~Classic ();
	Classic& operator= (const Classic& classic);
	virtual void Report() const; // reports all CD data
};

Classic::Classic (const char* songs_list, const char * s1, const char * s2, int n, double x)
	: Cd(s1, s2, n, x)
{
	songs = cpNewStr(songs_list);
}

Classic::Classic (const Classic& classic)
	: Cd(classic)
{
	songs = cpNewStr(classic.songs);
}

Classic::~Classic ()
{
	delete [] songs;
}

Classic & Classic::operator= (const Classic& classic)
{
	if (&classic == this)
		return (*this);
	Cd::operator=(classic);
	delete [] songs;
	songs = cpNewStr(classic.songs);
	return (*this);
}
```

```
void Classic::Report () const
{
	Cd::Report();
	cout << ", " << songs << endl;
}

void Bravo(const Cd & disk)
{
	disk.Report();
	cout << endl;
}

int main()
{
	Cd c1("Beatles", "Capitol", 14, 35.5);
	Classic c2 = Classic("Piano Sonata in B flat, Fantasia in C", "Alfred Brendel", "Philips", 2, 57.17);
	Cd *pcd = &c1;
	cout << "Using object directly:\n";
	c1.Report(); // use Cd method
	c2.Report(); // use Classic method
	cout << "Using type cd * pointer to objects:\n";
	pcd->Report(); // use Cd method for cd object
	pcd = &c2;
	pcd->Report(); // use Classic method for classic object
	cout << "Calling a function with a Cd reference argument:\n";
	Bravo(c1);
	Bravo(c2);
	cout << "Testing assignment: ";
	Classic copy;
	copy = c2;
	copy.Report();
}</cstring></iostream>
```

13.3 让三个类从一个基类 DMA 继承而来，然后用程序清单 13.10 对比测试，基类使用虚类。

```
//13.3 让三个类从一个基类 DMA 继承而来，然后用程序清单 13.10 对比测试，基类使用虚类。
#include <iostream>
#include <string>
using namespace std;

class DMA{
```

```
    string label;
    int rating;
public:
    DMA(const string l="null",int r=0)
    {
        label=l;
        rating=r;
    }
    virtual void test(){};
    virtual void tese2()    { cout<<"test2"; }
    DMA(const DMA &rs)
    {
        label=rs.label;
        rating=rs.rating;
    }
    virtual ~DMA() {}
    string lab() {return label;}
    int ra() {return rating;}
    friend ostream&operator<<(ostream &os,const DMA &rs)
    {
        os<<"label:"<<rs.label<<" rating:"<<rs.rating<<endl;="" return="" os;="" }="" virtual="" void="" show()="" {="" cout<<"label:"<<label<<"="" rating:"<<rating<<"="" "<<endl;="" };="" class="" basedma:public="" dma="" public:="" basedma(const="" string="" l="null" ,int="" r="0):DMA(l,r)" {}="" basedma(basedma="" &bd):dma(bd.lab(),bd.ra())="" test()const{};="" ~basedma(){}="" friend="" ostream&operator<<(ostream="" &os,const="" basedma="" &bd)="" os<<"this="" is="" basedma:="" ";="" os<<(dma="" &)bd;="" cout<<"this="" dma::show();="" cout<<endl;="" lacksdma:public="" color;="" lacksdma(const="" c="blank" ,const="" color="c;" ~lacksdma(){};="" 必须实现虚基类的所有虚函数="" lacksdma="" &ld)="" lacksdma:="" &)ld<<"="" color:"<<ld.color;="" 通过强制类型转换调用基类友元函数="" cout<<"="" color:"<<color;="" hasdma:public="" style;="" hasdma(const="" s="none" style="s;" ~hasdma(){};="" hasdma:="" style:"<<style;="" hasdma="" &hd)="" &)hd<<"="" style:"<<hd.style;="" int="" main="" ()="" *pd[3];="" 虚基类不能创建对象，但可以创建指向其的指针="" for(int="" i="0;i<3;i++)" cout<<"\nenter="" the="" label:";="" label;="" getline(cin,label,&amp;#39;\n&amp;#39;);="" rating:";="" rat;="" cin="">>rat;
        cout<<"Enter the 1 for baseDMA"<<endl <<"2="" for="" lacksdma"<<endl="" <<"3="" hasdma"<<endl;="" int="" temp;="" cin="">>temp;
        cin.get();
        if(temp==1)
            pd[i]=new baseDMA(label,rat);
        else if(temp==2)
        {
            cout<<"Enter the color:";
            string color;
            getline(cin,color);
            pd[i]=new lacksDMA(color,label,rat);
        }
        else if(temp==3)
```

```
        {
            cout<<"Enter the style:";
            string style;
            getline(cin,style);
            pd[i]=new hasDMA(style,label,rat);
        }
        else
        {
            cout<<"invalid input! try again!"<<endl; i--;="" }="" while(cin.get()!="'\n')" continue;="" cout<<endl;="" for(int="" i="0;i<3;i++)" pd[i]-="">show();
}</endl;></endl></rs.label<<"></string></iostream>
```

13.4 根据 Port 类派生出一个 VintagePort 类，完成并测试

```
//13.4 根据 Port 类派生出一个 VintagePort 类，完成并测试
#include <iostream>
#include <cstring>
using namespace std;

class Port
{
    char * brand;
    char style[20]; // i.e., tawny, ruby, vintage
    int bottles;
public:
    explicit Port(const char * br = "none", const char * st = "none", int b = 0);
    Port(const Port & p); // copy constructor
    virtual ~Port() { delete [] brand; }
    Port & operator=(const Port & p);
    virtual void Show() const;
    Port & operator+=(int b); // adds b to bottles
    Port & operator-=(int b); // subtracts b from bottles, if available
    int BottleCount() const { return bottles; }
    friend ostream & operator<<(ostream & os, const Port & p);
};

static char* cpNewStr (const char* p_src_txt)
{
    unsigned    str_len = strlen(p_src_txt);
    char*    p_des_txt = new char [str_len + 1];
    strcpy(p_des_txt, p_src_txt);
```

```
	return (p_des_txt);
}

static void cpStr (char* p_des_txt, const char* p_src_txt, unsigned des_arr_size)
{
	unsigned	str_len = strlen(p_src_txt) < des_arr_size-1 ? strlen(p_src_txt) : des_arr_size-1;
	strncpy(p_des_txt, p_src_txt, str_len);
	p_des_txt[str_len] = '\0';
}

Port::Port (const char * br, const char * st, int b)
	: brand(cpNewStr(br)), bottles(b)
{
	cpStr(style, st, 20);
}

Port::Port(const Port & p)
	: brand(cpNewStr(p.brand)), bottles(p.bottles)
{
	cpStr(style, p.style, 20);
}

void Port::Show() const
{
	cout	<< "Brand: " << brand << endl
		<< "Style: " << style << endl
		<< "Bottles: " << bottles << flush;
}

Port & Port::operator=(const Port & p)
{
	if (&p == this)
		return (*this);
	delete [] brand;
	brand = cpNewStr(p.brand);
	cpStr(style, p.style, 20);
	bottles = p.bottles;
	return (*this);
}

Port & Port::operator+=(int b)
{
	bottles += b;
	return (*this);
```

```
}

Port & Port::operator-=(int b)
{
	bottles -= b;
	return (*this);
}

ostream & operator<< (ostream & os, const Port & p)
{
	cout << p.brand << ", " << p.style << ", " << p.bottles << flush;
	return (os);
}

class VintagePort : public Port // style necessarily = "vintage"
{
	char * nickname; // i.e., "The Noble" or "Old Velvet", etc.
	int year; // vintage year
public:
	explicit VintagePort(const char * br = "", int b = 0, const char * nn = "", int y = 0);
	VintagePort(const VintagePort & vp);
	virtual ~VintagePort() { delete [] nickname; }
	VintagePort & operator=(const VintagePort & vp);
	virtual void Show() const;
	friend ostream & operator<<(ostream & os, const VintagePort & vp);
};

VintagePort::VintagePort (const char * br, int b, const char * nn, int y)
	: Port(br, "vintage", b), nickname(cpNewStr(nn)), year(y) {}

VintagePort::VintagePort (const VintagePort & vp)
	: Port(vp), nickname(cpNewStr(vp.nickname)), year(vp.year) {}

void VintagePort::Show () const
{
	Port::Show();
	cout << endl;
	cout << "Nickname: " << nickname << endl;
	cout << "Year: " << year << flush;
}

VintagePort & VintagePort::operator= (const VintagePort & vp)
{
	if (&vp == this)
```

```
            return (*this);
        Port::operator=(vp);
        delete [] nickname;
        nickname = cpNewStr(vp.nickname);
        year = vp.year;
        return (*this);
}

ostream & operator<< (ostream & os, const VintagePort & vp)
{
        os << Port(vp);
        cout << ", " << vp.nickname << ", " << vp.year << flush;
        return (os);
}

int main()
{
        Port        port1("gallo", "tawny", 20);
        cout << port1 << endl << endl;
        VintagePort vp("gallo", 24, "nice", 16);
        VintagePort vp2(vp);
        cout << vp2 << endl << endl;
        VintagePort vp3;
        vp3 = vp;
        cout << vp3 << endl << endl;
        Port*      p_port;
        p_port = &port1;
        p_port->Show();
        cout << endl;
        p_port = &vp;
        p_port->Show();
        cout << endl;
}</cstring></iostream>


//14
1、
//winec.h
#ifndef WINEC_H_
#define WINEC_H_

#include <iostream>
#include <string>
#include <valarray>
```

```
using namespace std;



template<class T1, class T2>
class Pair
{
private:
	T1 year;
	T2 bottles;
public:
	Pair(const T1 &yr, const T2 &bt) :year(yr), bottles(bt){}
	Pair(){}
	void Set(const T1 &yr, const T2 &bt);
	int Sum()const;
	void Show(int y)const;
};

template<class T1, class T2>
void Pair<T1,T2>::Set(const T1 &yr, const T2 &bt)
{
	year = yr;
	bottles = bt;
}
template<class T1,class T2>
int Pair<T1, T2>::Sum()const
{
	returnbottles.sum();
}

template<class T1, class T2>
void Pair<T1,T2>::Show(int y)const
{
	for (inti = 0; i< y; i++)
		cout<< "\t" << year[i] << "\t" << bottles[i] <<endl;
}

typedefvalarray<int>ArrayInt;
typedef Pair<ArrayInt, ArrayInt>PairArray;

class Wine
{
private:
```

```
        PairArrayyb;
        stringfullname;
        intyrs;
public:
        Wine(){}
        Wine(const char *l, int y, constintyr[], constint bot[]);
        Wine(const char *l, int y);
        voidGetBottles();
        string&Label();
        void Show()const;
        int sum()const;
};

#endif
//winec.cpp
#include "winec.h"

Wine::Wine(const char *l, int y, constintyr[], constint bot[])
{
        fullname = l;
        yrs = y;
        yb.Set(ArrayInt(yr, yrs), ArrayInt(bot, yrs));
}
Wine::Wine(const char *l, int y)
{
        fullname = l;
        yrs = y;
}
void Wine::GetBottles()
{
        ArrayIntyr(yrs), bt(yrs);
        for (inti = 0; i<yrs; i++)
        {
                cout<< "Enter the year: ";
                cin>>yr[i];
                cout<< "Enter the bottles: ";
                cin>>bt[i];
        }
        while (cin.get() != '\n')
                continue;
        yb.Set(yr, bt);
}
string&Wine::Label()
{
```

```
        returnfullname;
}

void Wine::Show()const
{
        cout<< "Wine: " <<fullname<<endl;
        cout<< "\tYear\tBottles\n";
        yb.Show(yrs);
}

int Wine::sum()const
{
        returnyb.Sum();
}
//main.cpp
#include "winec.h"

int main(void)
{
        cout<< "Enter name of wine: ";
        char lab[50];
        cin.getline(lab, 50);
        cout<< "Enter number of years: ";
        intyrs;
        cin>>yrs;
        Wine holding(lab, yrs);
        holding.GetBottles();
        holding.Show();
        constint YRS = 3;
        int y[YRS] = { 1993, 1995, 1998 };
        int b[YRS] = { 48, 60, 72 };
        Wine more("Gushing Grape Red", YRS, y, b);
        more.Show();
        cout<< "Total bottles for " <<more.Label()
                << ": " <<more.sum() <<endl;
        cout<< "Bye\n";
        system("pause");
        return 0;
}

2、
//winec.h
#ifndef WINEC_H_
#define WINEC_H_
```

```
#include <iostream>
#include <string>
#include <valarray>

using namespace std;

template<class T1, class T2>
class Pair
{
private:
	T1 year;
	T2 bottles;
public:
	Pair(const T1 &yr, const T2 &bt) :year(yr), bottles(bt){}
	Pair(){}
	void Set(const T1 &yr, const T2 &bt);
	int Sum()const;
	void Show(int y)const;
};

template<class T1, class T2>
void Pair<T1,T2>::Set(const T1 &yr, const T2 &bt)
{
	year = yr;
	bottles = bt;
}
template<class T1,class T2>
int Pair<T1, T2>::Sum()const
{
	returnbottles.sum();
}

template<class T1, class T2>
void Pair<T1,T2>::Show(int y)const
{
	for (inti = 0; i< y; i++)
		cout<< "\t" << year[i] << "\t" << bottles[i] <<endl;
}

typedefvalarray<int>ArrayInt;
typedef Pair<ArrayInt, ArrayInt>PairArray;

class Wine :private PairArray, private string
```

```
{
private:
	intyrs;
public:
	Wine(){}
	Wine(const char *l, int y, constintyr[], constint bot[]);
	Wine(const char *l, int y);
	voidGetBottles();
	string&Label();
	void Show()const;
	int sum()const;
};

#endif
//winec.cpp
#include "winec.h"

Wine::Wine(const char *l, int y, constintyr[], constint bot[]) :string(l), yrs(y), PairArray(ArrayInt(yr,
y), ArrayInt(bot, y))
{

}
Wine::Wine(const char *l, int y) : string(l), yrs(y)
{

}
void Wine::GetBottles()
{
	ArrayIntyr(yrs), bt(yrs);
	for (inti = 0; i<yrs; i++)
	{
		cout<< "Enter the year: ";
		cin>>yr[i];
		cout<< "Enter the bottles: ";
		cin>>bt[i];
	}
	while (cin.get() != '\n')
		continue;
	PairArray::Set(yr, bt);
}
string&Wine::Label()
{
	return (string &)(*this);
}
```

```
void Wine::Show()const
{
	cout<< "Wine: " << (string &)(*this) <<endl;
	cout<< "\tYear\tBottles\n";
	PairArray::Show(yrs);
}

int Wine::sum()const
{
	returnPairArray::Sum();
}
//main.cpp
#include "winec.h"

int main(void)
{
	cout<< "Enter name of wine: ";
	char lab[50];
	cin.getline(lab, 50);
	cout<< "Enter number of years: ";
	intyrs;
	cin>>yrs;
	Wine holding(lab, yrs);
	holding.GetBottles();
	holding.Show();
	constint YRS = 3;
	int y[YRS] = { 1993, 1995, 1998 };
	int b[YRS] = { 48, 60, 72 };
	Wine more("Gushing Grape Red", YRS, y, b);
	more.Show();
	cout<< "Total bottles for " <<more.Label()
		<< ": " <<more.sum() <<endl;
	cout<< "Bye\n";
	system("pause");
	return 0;
}

3、
//queuetp.h
#ifndef QUEUETP_H_
#define QUEUETP_H_

#include <iostream>
```

```
#include <string>
#include <cstring>
using namespace std;

template<typename T>
classQueueTp
{
private:
	struct Node { T item; struct Node *next; };
	Node *front;
	Node *rear;
	int items;
	constintqsize;
	QueueTp(constQueueTp&q) :qsize(0){}
	QueueTp&operator=(constQueueTp&q){ return *this; }
public:
	QueueTp(intqs = 10);
	~QueueTp();
	boolisempty()const;
	boolisfull()const;
	intqueuecount()const;
	boolenqueue(const T &item);
	booldequeue(T &item);
};

template<typename T>
QueueTp<T>::QueueTp(intqs) :qsize(qs)
{
	front = rear = NULL;
	items = 0;
}

template<typename T>
QueueTp<T>::~QueueTp()
{
	Node *temp;
	while (front != NULL)
	{
		temp = front;
		front = front->next;
		delete temp;
	}
}
```

```
template<typename T>
boolQueueTp<T>::isempty()const
{
	return items == 0;
}

template<typename T>
boolQueueTp<T>::isfull()const
{
	return items == qsize;
}

template<typename T>
intQueueTp<T>::queuecount()const
{
	return items;
}

template<typename T>
boolQueueTp<T>::enqueue(const T &item)
{
	if (isfull())
		return false;
	Node *add = new Node;
	add->item = item;
	add->next = NULL;
	items++;
	if (front == NULL)
		front = add;
	else
		rear->next = add;
	rear = add;
	return true;
}

template<typename T>
boolQueueTp<T>::dequeue(T &item)
{
	if (front == NULL)
		return false;
	item = front->item;
	items--;
	Node *temp = front;
```

```
        front = front->next;
        delete temp;
        if (items == 0)
                rear = NULL;
        return true;
}

class Worker
{
private:
        stringfullname;
        long id;

public:
        Worker() :fullname("no one"), id(0L){}
        Worker(const string &s, long n) :fullname(s), id(n){}
        ~Worker();
        void Set();
        void Show()const;
};


#endif
//workermi.cpp
#include "queuetp.h"

Worker::~Worker(){}

void Worker::Show()const
{
        cout<< "Name: " <<fullname<<endl;
        cout<< "Employee ID: " << id <<endl;
}

void Worker::Set()
{
        cout<< "Enter worker's name: ";
        getline(cin, fullname);
        cout<< "Enter worker's ID: ";
        cin>> id;
        while (cin.get() != '\n')
                continue;
}
//main.cpp
```

```
#include "queuetp.h"

constint Size = 5;

int main()
{

	QueueTp<Worker *>lolas(Size);
	Worker *temp;
	intct;
	for (ct = 0; ct< Size; ct++)
	{
		charch;
		cout<< "Enter the command:\n"
			<< "A or a enter queue, "
			<< "P or p delete queue, "
			<< "Q or q quit.\n";
		cin>>ch;
		while (strchr("apq", ch) == NULL)
		{
			cout<< "Please enter a p or q: ";
			cin>>ch;
		}
		if (ch == 'q')
			break;

		switch(ch)
		{
		case'a':
			temp = new Worker;
			cin.get();
			temp->Set();
			if (lolas.isfull())
				cout<< "Queue already full\n";
			else
				lolas.enqueue(temp);
			break;
		case'p':
			if (lolas.isempty())
				cout<< "Queue already empty\n";
			else
				lolas.dequeue(temp);
			break;
		}
```

```
        }
        cout<< "\nHere the total count: ";
        cout<<lolas.queuecount();
        cout<< "Done.\n";
        system("pause");
        return 0;
}
```

4、

```
//person.h
#ifndef PERSON_H_
#define PERSON_H_

#include <iostream>
#include <string>
#include <cstdlib>
#include <cstring>
using namespace std;
class Person
{
private:
        stringfirstname;
        stringlastname;
protected:
        virtual void Data()const;
        virtual void Get();
public:
        Person():firstname("no one"),lastname("no one"){}
        Person(const string &f,const string &l):firstname(f),lastname(l){}
        Person(const Person &p):Person(p){}
        virtual ~Person() = 0;
        virtual void Set() = 0;
        virtual void Show()const = 0;
};

classGunslinger:virtual public Person
{
private:
        intnumsk;
protected:
        void Data()const;
        void Get();
public:
```

```
    Gunslinger():numsk(0),Person(){}
    Gunslinger(intnk, const string &f, const string &l) :numsk(nk), Person(f, l){}
    Gunslinger(intnk, const Person &p):numsk(nk),Person(p){}
    void Show()const;
    void Set();
    double Draw()const;
};

classPokerPlayer:virtual public Person
{
protected:
    void Data()const;
public:
    PokerPlayer():Person(){}
    PokerPlayer(const string &f, const string &l) : Person(f, l){}
    PokerPlayer(const Person &p):Person(p){}
    int Draw()const;
    void Show()const;
    void Set(){ Person::Set(); }
};

classBadDude:publicGunslinger,publicPokerPlayer
{
protected:
    void Data()const;
    void Get();
public:
    BadDude(){}
    BadDude(intnk , const string &f, const string &l)
        :Person(f, l), Gunslinger(nk, f, l), PokerPlayer(f, l){}
    BadDude(intnk, const Person &p)
        :Person(p), Gunslinger(nk, p), PokerPlayer(p){}
    BadDude(const Gunslinger &g)
        :Person(g),Gunslinger(g),PokerPlayer(g){}
    BadDude(intnk, constPokerPlayer&po)
        :Person(po), Gunslinger(nk, po), PokerPlayer(po){}
    doubleGdraw()const;
    intCdraw()const;
    void Set();
    void Show()const;
};

#endif
//person.cpp
```

```
#include "person.h"

Person::~Person(){}

void Person::Data()const
{
	cout<< "First name is : " <<firstname<<endl;
	cout<< "Last name is : " <<lastname<<endl;
}
void Person::Get()
{
	cout<< "Enter first name: \n";
	getline(cin, firstname);
	cout<< "Enter last name: \n";
	getline(cin, lastname);
}

void Person::Show()const
{
	Data();
}

void Person::Set()
{
	Get();
}

void Gunslinger::Data()const
{
	cout<< "Nick is :" <<numsk<<endl;
	cout<< "The time of get the gun :" << Gunslinger::Draw() <<endl;
}
void Gunslinger::Get()
{
	cout<< "Enter Nick: \n";
	cin>>numsk;
}

void Gunslinger::Set()
{
	cout<< "Enter Guns name: \n";
	Person::Get();
	Get();
}
```

```
void Gunslinger::Show()const
{
	cout<< "Gunslinger: \n";
	Person::Data();
	Data();
}

double Gunslinger::Draw()const
{
	return rand() % 3 + 1;
}

intPokerPlayer::Draw()const
{
	return rand() % 52 + 1;
}

voidPokerPlayer::Data()const
{
	cout<< "The cards :" << Draw() <<endl;
}

voidPokerPlayer::Show()const
{
	cout<< "PokerPlayer :\n";
	Person::Data();
	Data();
}

doubleBadDude::Gdraw()const
{
	return Gunslinger::Draw();
}
intBadDude::Cdraw()const
{
	returnPokerPlayer::Draw();
}

voidBadDude::Data()const
{
	Gunslinger::Data();
	PokerPlayer::Data();
	cout<< "The next cards: " <<Cdraw() <<endl;
```

```
        cout<< "The time of BadDude get the gun: " <<Gdraw() <<endl;
}
voidBadDude::Get()
{
        Gunslinger::Get();
}

voidBadDude::Set()
{
        cout<< "Enter BadDude name: \n";
        Person::Get();
        Get();
}
voidBadDude::Show()const
{
        cout<< "BadDude: \n";
        Person::Data();
        Data();
}
//main.cpp
#include "person.h"

constint Size=5;
int main ()
{
        Person *per[Size];
        intct;
        for (ct = 0; ct< Size; ct++)
        {
                char choice;
                cout<< "Enter the Person: \n"
                        << "g: gunslinger      p: poker    "
                        << "b: bad dude      q: quit\n";
                cin>> choice;
                while (strchr("gpbq", choice) == NULL)
                {
                        cout<< "Please enter a p,g,o,q: ";
                        cin>> choice;
                }
                if (choice == 'q')
                        break;
                switch (choice)
                {
                case'g':
```

```
                per[ct] = new Gunslinger;
                break;
            case'p':
                per[ct] = new PokerPlayer;
                break;
            case'b':
                per[ct] = new BadDude;
                break;
            }
            cin.get();
            per[ct]->Set();
        }
        cout<< "\nHere is your staff:\n";
        inti;
        for (i = 0; i<ct; i++)
        {
            cout<<endl;
            per[i]->Show();
        }
        for (i = 0; i<ct; i++)
            delete per[i];
        cout<< "Bye\n";
        system("pause");
        return 0;
}
```

5、

```
//emp.h
#ifndef EMP_H_
#define EMP_H_

#include <iostream>
#include <string>
using namespace std;

classabstr_emp
{
private:
        stringfname;
        stringlname;
        string job;
public:
        abstr_emp();
        abstr_emp(const string &fn, const string &ln,
```

```
            const string &j);
        virtual void ShowAll()const;
        virtual void SetAll();
        friendostream&operator<<(ostream&os, constabstr_emp&e);
        virtual ~abstr_emp() = 0;
};

class employee :public abstr_emp
{
public:
        employee();
        employee(const string &fn, const string &ln,
            const string &j);
        virtual void ShowAll()const;
        virtual void SetAll();
};

class manager :virtual public abstr_emp
{
private:
        intinchargeof;
protected:
        intInChargeOf()const { return inchargeof; }
        int&InChargeOf(){ return inchargeof; }
public:
        manager();
        manager(const string &fn, const string &ln,
            const string &j, intico = 0);
        manager(constabstr_emp&e, intico = 0);
        manager(const manager &m);
        virtual void ShowAll()const;
        virtual void SetAll();
        voidgetInCharge(){
            cout<< "Enter inchargeof: ";
            cin>>inchargeof;
        }
};

class fink :virtual public abstr_emp
{
private:
        stringreportsto;
protected:
        const string ReportsTo()const{ return reportsto; }
```

```
        string&ReportsTo(){ return reportsto; }
public:
        fink();
        fink(const string &fn, const string &ln,
                const string &j, const string &rpo);
        fink(constabstr_emp&e, const string &rpo);
        fink(const fink &e);
        virtual void ShowAll()const;
        virtual void SetAll();
        voidgetReportsTo(){
                cout<< "Enter reportsto: ";
                cin>>reportsto;
        }
};

classhighfink :public manager, public fink
{
public:
        highfink();
        highfink(const string &fn, const string &ln,
                const string &j, const string &rpo, intico = 0);
        highfink(constabstr_emp&e, const string &rpo, intico = 0);
        highfink(const fink &f, intico = 0);
        highfink(const manager &m, const string &rpo);
        highfink(consthighfink&h);
        virtual void ShowAll()const;
        virtual void SetAll();
};


#endif
//emp.cpp
#include "emp.h"

abstr_emp::abstr_emp() :fname("no one"), lname("no one"), job("no job")
{

}
abstr_emp::abstr_emp(const string &fn, const string &ln,
        const string &j) : fname(fn), lname(ln), job(j)
{

}
voidabstr_emp::ShowAll()const
```

```
{
	cout<< "Firstname: " <<fname<<endl;
	cout<< "Lastname: " <<lname<<endl;
	cout<< "Job is: " << job <<endl;

}
voidabstr_emp::SetAll()
{
	cout<< "Enter firstname: ";
	getline(cin, fname);
	cout<< "Enter lastname: ";
	getline(cin, lname);
	cout<< "Enter position: ";
	getline(cin, job);
}
ostream&operator<<(ostream&os, constabstr_emp&e)
{
	os<<e.fname<< " " <<e.lname<< ", " <<e.job<<endl;
	returnos;
}
abstr_emp::~abstr_emp()
{

}


employee::employee() :abstr_emp()
{

}
employee::employee(const string &fn, const string &ln,
	const string &j) : abstr_emp(fn, ln, j)
{

}
void employee::ShowAll()const
{
	abstr_emp::ShowAll();
}
void employee::SetAll()
{
	abstr_emp::SetAll();
}
```

```
manager::manager() :abstr_emp()
{

}
manager::manager(const string &fn, const string &ln,
	const string &j, intico) : abstr_emp(fn, ln, j), inchargeof(ico)
{

}
manager::manager(constabstr_emp&e, intico) : abstr_emp(e), inchargeof(ico)
{

}
manager::manager(const manager &m) : abstr_emp(m)
{

}
void manager::ShowAll()const
{
	abstr_emp::ShowAll();
	cout<< "Inchargeof: " <<InChargeOf() <<endl;
}
void manager::SetAll()
{
	abstr_emp::SetAll();
	cout<< "Enter inchargeof: ";
	(cin>>inchargeof).get();
}


fink::fink() :abstr_emp()
{

}
fink::fink(const string &fn, const string &ln,
	const string &j, const string &rpo) : abstr_emp(fn, ln, j), reportsto(rpo)
{

}
fink::fink(constabstr_emp&e, const string &rpo) : abstr_emp(e), reportsto(rpo)
{

}
```

```
fink::fink(const fink &e) : abstr_emp(e)
{

}
void fink::ShowAll()const
{
	abstr_emp::ShowAll();
	cout<< "Reportsto: " <<ReportsTo() <<endl;
}
void fink::SetAll()
{
	abstr_emp::SetAll();
	cout<< "Enter reportsto: ";
	cin>>reportsto;
}


highfink::highfink() :abstr_emp(), manager(), fink()
{

}
highfink::highfink(const string &fn, const string &ln,
	const string &j, const string &rpo, intico) : abstr_emp(fn, ln, j), manager(fn, ln, j, ico), fink(fn,
ln, j, rpo)
{

}
highfink::highfink(constabstr_emp&e, const string &rpo, intico) : abstr_emp(e), manager(e, ico),
fink(e, rpo)
{

}
highfink::highfink(const fink &f, intico) : abstr_emp(f), manager(f, ico), fink(f)
{

}
highfink::highfink(const manager &m, const string &rpo) : abstr_emp(m), manager(m), fink(m,
rpo)
{

}
highfink::highfink(consthighfink&h) : abstr_emp(h), manager(h), fink(h)
{
```

```
}
voidhighfink::ShowAll()const
{
	abstr_emp::ShowAll();
	cout<< "InChargeOf: " << manager::InChargeOf() <<endl;
	cout<< "ReportsTo: " << fink::ReportsTo() <<endl;
}
voidhighfink::SetAll()
{
	abstr_emp::SetAll();
	manager::getInCharge();
	fink::getReportsTo();
}
//useemp.cpp
#include "emp.h"

int main(void)
{
	employeeem("Trip", "Harris", "Thumper");
	cout<<em<<endl;
	em.ShowAll();
	manager ma("Amorphia", "Spindragon", "Nuancer", 5);
	cout<< ma <<endl;
	ma.ShowAll();

	fink fi("Matt", "Oggs", "Oiler", "Juno Barr");
	cout<< fi <<endl;
	fi.ShowAll();
	highfinkhf(ma, "Curly Kew");
	hf.ShowAll();
	cout<< "Press a key for next phase:\n";
	cin.get();
	highfink hf2;
	hf2.SetAll();

	cout<< "Using an abstr_emp * pointer:\n";
	abstr_emp *tri[4] = { &em, &fi, &hf, &hf2 };
	for (inti = 0; i< 4; i++)
		tri[i]->ShowAll();
	system("pause");
	return 0;
}
```

```
//15
1、
//tv.h
#ifndef TV_H_
#define TV_H_

#include <iostream>
using namespace std;

classTv
{
    friend class Remote;
public:

    enum { Off, On };
    enum { MinVal, MaxVal = 20 };
    enum { Antenna, Cable };
    enum { TV, DVD };
    enum { USUAL, EXCHANGE };

    Tv(int s = Off, int mc = 125) :state(s), volume(5),
        maxchannel(mc), channel(2), mode(Cable), input(TV){}
    ~Tv(){}
    voidonoff(){ state = (state == On) ? Off : On; }
    boolison()const{ return state == On; }
    boolvolup();
    boolvoldown();
    voidchanup();
    voidchandown();
    voidset_mode(){ mode = (mode == Antenna) ? Cable : Antenna; }
    voidset_input(){ input = (input == TV) ? DVD : TV; }
    void settings()const;
    voidset_rmode(Remote &r);
private:
    int state;
    int volume;
    intmaxchannel;
    int channel;
    int mode;
    int input;
};

class Remote
{
```

```
private:
	friend class Tv;
	enum { USUAL, EXCHANGE };
	int mode;
	intfmode;
public:
	Remote(int m = Tv::TV, int f = USUAL) :mode(m), fmode(f){}
	boolvolup(Tv&t){ return t.volup(); }
	boolvoldown(Tv&t){ return t.voldown(); }
	voidonoff(Tv&t){ t.onoff(); }
	voidchanup(Tv&t){ t.chanup(); }
	voidchandown(Tv&t){ t.chandown(); }
	voidset_chan(Tv&t, int c){ t.channel = c; }
	voidset_mode(Tv&t){ t.set_mode(); }
	voidset_input(Tv&t){ t.set_input(); }
	voidmode_show()const{ cout<< "Remote pretent mode is " <<fmode<<endl; }
};

inline void Tv::set_rmode(Remote &r)
{
	if (ison())
	{
		r.fmode = Remote::EXCHANGE;
		r.mode_show();
	}
}

#endif
//tvfm.h
#ifndef TVFM_H_
#define TVFM_H_
#include <iostream>
using namespace std;

classTv;

class Remote
{
public:
	enum State{ Off, On };
	enum { MinVal, MaxVal = 20 };
	enum { Antenna, Cable };
	enum { TV, DVD };
private:
```

```
    int mode;
public:
    Remote(int m = TV) :mode(m){}
    boolvolup(Tv&t);
    boolvoldown(Tv&t);
    voidonoff(Tv&t);
    voidchanup(Tv&t);
    voidchandown(Tv&t);
    voidset_chan(Tv&t, int c);
    voidset_mode(Tv&t);
    voidset_input(Tv&t);
};

classTv
{
public:
    friend void Remote::set_chan(Tv&t, int c);
    enum State{ Off, On };
    enum { MinVal, MaxVal = 20 };
    enum { Antenna, Cable };
    enum { TV, DVD };

    Tv(int s = Off, int mc = 125) :state(s), volume(5),
        maxchannel(mc), channel(2), mode(Cable), input(TV){}
    ~Tv(){}
    voidonoff(){ state = (state == On) ? Off : On; }
    boolison()const{ return state == On; }
    boolvolup();
    boolvoldown();
    voidchanup();
    voidchandown();
    voidset_mode(){ mode = (mode == Antenna) ? Cable : Antenna; }
    voidset_input(){ input = (input == TV) ? DVD : TV; }
    void settings()const;

private:
    int state;
    int volume;
    intmaxchannel;
    int channel;
    int mode;
    int input;
};
```

```
inlinebool Remote::volup(Tv&t){ return t.volup(); }
inlinebool Remote::voldown(Tv&t){ return t.voldown(); }
inline void Remote::onoff(Tv&t){ t.onoff(); }
inline void Remote::chanup(Tv&t){ t.chanup(); }
inline void Remote::chandown(Tv&t){ t.chandown(); }
inline void Remote::set_chan(Tv&t, int c){ t.channel = c; }
inline void Remote::set_mode(Tv&t){ t.set_mode(); }
inline void Remote::set_input(Tv&t){ t.set_input(); }

#endif
//tv.cpp
#include "tv.h"
//#include "tvfm.h"
boolTv::volup()
{
	if (volume <MaxVal)
	{
		volume++;
		return true;
	}
	else
		return false;
}
boolTv::voldown()
{
	if (volume>MinVal)
	{
		volume--;
		return true;
	}
	else
		return false;
}
voidTv::chanup()
{
	if (channel <maxchannel)
		channel++;
	else
		channel = 1;
}
voidTv::chandown()
{
	if (channel>1)
		channel--;
```

```
        else
                channel = maxchannel;
}
voidTv::settings()const
{
        cout<< "TV is " << (state == Off ? "Off" : "On") <<endl;
        if (state == On)
        {
                cout<< "Volume setting = " << volume <<endl;
                cout<< "Channel setting = " << channel <<endl;
                cout<< "Mode = " << (mode == Antenna ? "antenna" : "cable") <<endl;
                cout<< "Input = " << (input == TV ? "TV" : "DVD") <<endl;
        }
}
//use_tv.cpp
#include "tv.h"
int main()
{
        Tv s42;
        Remote grey;
        grey.mode_show();
        cout<< "Initial settings for 42\" TV:\n";
        s42.settings();
        s42.onoff();
        s42.chanup();
        cout<< "\nAdjusted settings for 42\" TV:\n";
        s42.settings();


        s42.set_rmode(grey);
        grey.set_chan(s42, 10);
        grey.volup(s42);
        grey.volup(s42);
        cout<< "\n42\" settings after using remote:\n";
        s42.settings();

        Tvs58(Tv::On);
        s58.set_mode();
        grey.set_chan(s58, 28);
        cout<< "\n58\" settings:\n";
        s58.settings();
        s58.set_rmode(grey);
        system("pause");
        return 0;
```

```
}

2、
//exc_mean.h
#ifndef EXC_MEAN_H_
#define EXC_MEAN_H_
#include <iostream>
#include <cmath>
#include <stdexcept>
#include <string>
using namespace std;

classbad_hmean :public logic_error
{
private:
	string name;
public:
	explicitbad_hmean(const string &n = "hmean", const string &s = "Error in hmean()\n");
	stringmesg();
	virtual ~bad_hmean()throw(){}
};

bad_hmean::bad_hmean(const string &n, const string &s) :name(n), logic_error(s)
{

}

inline string bad_hmean::mesg()
{
	return "hmean() arguments a=-b should be div a+b=0!\n";
}

classbad_gmean :public logic_error
{
private:
	string name;
public:
	explicitbad_gmean(const string &n = "gmean", const string &s = "Error in gmean()\n");
	stringmesg();
	virtual ~bad_gmean()throw(){}
};

bad_gmean::bad_gmean(const string &n, const string &s) :name(n), logic_error(s)
{
```

```
}

inline string bad_gmean::mesg()
{
	return "gmean() arguments should be >= 0\n";
}

#endif
//error.cpp
#include "exc_mean.h"

doublehmean(double a, double b);
doublegmean(double a, double b);

int main()
{
	double x, y, z;
	cout<< "Enter two numbers:";
	while (cin>> x >> y)
	{
		try{
			z = hmean(x, y);
			cout<< "Harmonic mean of " << x << " and " << y
				<<" is " << z <<endl;
			cout<< "Geomettric mean of " << x << " and " << y
				<<" is " <<gmean(x, y) <<endl;
			cout<< "Enter next set of numbers <q to quit>: ";
		}
		catch (bad_hmean&bg)
		{
			cout<<bg.what();
			cout<< "Error message: \n" <<bg.mesg() <<endl;
			cout<< "Try again.\n";
			continue;
		}
		catch (bad_gmean&hg)
		{
			cout<<hg.what();
			cout<< "Error message: \n" <<hg.mesg() <<endl;
			cout<< "Sorry, you don't get to play and more.\n";
			break;
		}
	}
```

```
        cout<< "Bye!\n";
        system("pause");
        return 0;
}

doublehmean(double a, double b)
{
        if (a == -b)
                throwbad_hmean();
        return 2.0*a*b / (a + b);
}
doublegmean(double a, double b)
{
        if (a < 0 || b < 0)
                throwbad_gmean();
        returnsqrt(a*b);
}
```

3、

```
//exc_mean.h
#ifndef EXC_MEAN_H_
#define EXC_MEAN_H_
#include <iostream>
#include <cmath>
#include <stdexcept>
#include <string>
using namespace std;

classbad_hmean :public logic_error
{
private:

        string name;
public:
        double v1;
        double v2;
        explicitbad_hmean(double a = 0, double b = 0,
                const string &s = "Error in hmean()\n");
        voidmesg();
        virtual ~bad_hmean()throw(){}
};

bad_hmean::bad_hmean(double a, double b, const string &s)
:v1(a), v2(b), logic_error(s)
```

```
{
	name = "hmean";
}

inline void bad_hmean::mesg()
{
	cout<< name << "(" << v1 << ", " << v2
		<< ") arguments a=-b should be div a+b=0!\n";
}

classbad_gmean :public bad_hmean
{
private:
	string name;
public:
	explicitbad_gmean(double a = 0, double b = 0,
		const string &s = "Error in gmean()\n");
	voidmesg();
	virtual ~bad_gmean()throw(){}
};

bad_gmean::bad_gmean(double a, double b, const string &s)
:bad_hmean(a, b, s)
{
	name = "gmean";
}

inline void bad_gmean::mesg()
{
	cout<< name << "(" <<bad_hmean::v1 << ", " <<bad_hmean::v2 << ") arguments should
be >= 0\n";
}

#endif
//error.cpp
#include "exc_mean.h"

doublehmean(double a, double b);
doublegmean(double a, double b);

int main()
{
	double x, y, z;
	cout<< "Enter two numbers:";
```

```
	while (cin>> x >> y)
	{
		try{
			z = hmean(x, y);
			cout<< "Harmonic mean of " << x << " and " << y
				<<" is " << z <<endl;
			cout<< "Geomettric mean of " << x << " and " << y
				<<" is " <<gmean(x, y) <<endl;
			cout<< "Enter next set of numbers <q to quit>: ";
		}
		catch (bad_gmean&hg)
		{
			cout<<hg.what();
			cout<< "Error message: \n";
			hg.mesg();
			cout<<endl;
			cout<< "Sorry, you don't get to play and more.\n";
			break;
		}
		catch (bad_hmean&bg)
		{
			cout<<bg.what();
			cout<< "Error message: \n";
			bg.mesg();
			cout<<endl;
			cout<< "Try again.\n";
			continue;
		}

	}
	cout<< "Bye!\n";
	system("pause");
	return 0;
}

doublehmean(double a, double b)
{
	if (a == -b)
		throwbad_hmean();
	return 2.0*a*b / (a + b);
}
doublegmean(double a, double b)
{
	if (a < 0 || b < 0)
```

```
        throwbad_gmean();
    returnsqrt(a*b);
}

4、
//sales.h
#ifndef SALES_H_
#define SALES_H_
#include <stdexcept>
#include <string>
#include <cstring>
#include <iostream>
#include <cstdlib>
using namespace std;

class Sales
{
public:
    enum { MONTHS = 12 };
    classbad_index:publiclogic_error
    {
    private:
        int bi;
    public:
        explicitbad_index(int ix, const string &s = "Index error in Sales object\n");
        intbi_val()const { return bi; }
        virtual ~bad_index()throw(){}
    };
    explicit Sales(intyy = 0);
    Sales(intyy, const double *gr, int n);
    virtual ~Sales(){}
    int Year()const { return year; }
    virtual double operator[](inti)const;
    virtual double &operator[](inti);
private:
    double gross[MONTHS];
    int year;
};

classLabeledSales :public Sales
{
public:
    classnbad_index :public Sales::bad_index
    {
```

```
        private:
                std::stringlbl;
        public:
                nbad_index(const string &lb, int ix,
                        const string &s = "Index error in LabeledSales object\n");
                const string &label_val()const { return lbl; }
                virtual ~nbad_index()throw(){}
        };
        explicitLabeledSales(const string &lb = "none", intyy = 0);
        LabeledSales(const string &lb, intyy, const double *gr, int n);
        virtual ~LabeledSales(){}
        const string &Label()const { return label; }
        virtual double operator[](inti)const;
        virtual double &operator[](inti);
private:
        string label;
};

#endif
//sales.cpp
#include "sales.h"

Sales::bad_index::bad_index(int ix,
        const string &s) :logic_error(s), bi(ix)
{

}

Sales::Sales(intyy)
{
        year = yy;
        for (inti = 0; i< MONTHS; ++i)
                gross[i] = 0;
}
Sales::Sales(intyy, const double *gr, int n)
{
        year = yy;
        intlim = (n < MONTHS) ? n : MONTHS;
        inti;
        for (i = 0; i<lim; ++i)
                gross[i] = gr[i];
        for (; i< MONTHS; ++i)
                gross[i] = 0;
}
```

```
double Sales::operator[](inti)const
{
	if (i< 0 || i>= MONTHS)
		throwbad_index(i);
	return gross[i];
}
double&Sales::operator[](inti)
{
	if (i< 0 || i>= MONTHS)
		throwbad_index(i);
	return gross[i];
}

LabeledSales::nbad_index::nbad_index(const string &lb, int ix,
	const string &s) :Sales::bad_index(ix, s)
{
	lbl = lb;
}

LabeledSales::LabeledSales(const string &lb, intyy) : Sales(yy)
{
	label = lb;
}
LabeledSales::LabeledSales(const string &lb, intyy, const double *gr, int n) : Sales(yy, gr, n)
{
	label = lb;
}

doubleLabeledSales::operator[](inti)const
{
	if (i< 0 || i>= MONTHS)
		thrownbad_index(Label(), i);
	return Sales::operator[](i);
}
double&LabeledSales::operator[](inti)
{
	if (i< 0 || i>= MONTHS)
		thrownbad_index(Label(), i);
	return Sales::operator[](i);
}
//use_sales.cpp
#include "sales.h"

int main()
```

```
{
    double vals1[12] =
    {
        1220, 1100, 1122, 2212, 1232, 2334,
        2884, 2393, 3302, 2922, 3002, 3544
    };
    double vals2[12] =
    {
        12, 11, 22, 21, 32, 24,
        28, 29, 33, 29, 32, 35
    };
    Sales sales1(2011, vals1, 12);
    LabeledSalessales2("Blogstar", 2012, vals2, 12);
    Sales::bad_index *s;
    LabeledSales::nbad_index *l;
    cout<< "First try block:\n";
    try
    {
        inti;
        cout<< "Year = " << sales1.Year() <<endl;
        for (i = 0; i< 12; ++i)
        {
            cout<< sales1[i] << ' ';
            if (i % 6 == 5)
                cout<<endl;
        }
        cout<< "Year = " << sales2.Year() <<endl;
        cout<< "Label = " << sales2.Label() <<endl;
        for (i = 0; i<= 12; ++i)
        {
            cout<< sales2[i] << ' ';
            if (i % 6 == 5)
                cout<<endl;
        }
        cout<< "End of try block 1.\n";
    }
    catch (logic_error&bad)
    {
        cout<<bad.what();
        if (l = dynamic_cast<LabeledSales::nbad_index *>(&bad))
        {
            cout<< "Comany: " << l->label_val() <<endl;
            cout<< "bad index: " << l->bi_val() <<endl;
        }
```

```
        else if (s = dynamic_cast<Sales::bad_index *>(&bad))
            cout<< "bad index: " << s->bi_val() <<endl;
    }
    cout<< "\nNext try block:\n";
    try
    {
        sales2[2] = 37.5;
        sales1[20] = 23345;
        cout<< "End of try block 2.\n";
    }
    catch (logic_error&bad)
    {
        cout<<bad.what();
        if (l = dynamic_cast<LabeledSales::nbad_index *>(&bad))
        {
            cout<< "Comany: " << l->label_val() <<endl;
            cout<< "bad index: " << l->bi_val() <<endl;
        }
        else if (s = dynamic_cast<Sales::bad_index *>(&bad))
            cout<< "bad index: " << s->bi_val() <<endl;
    }
    cout<< "done\n";
    system("pause");
    return 0;
}
```

Chapter 16

PE 16-1

```
// pe16-1.cpp -- one of many possible solutions
#include <iostream>
#include <string>

bool isPal(const std::string & s);

int main()
{
    std::string input;
```

```
    std::cout << "Enter a string (empty string to quit):\n";
    std::getline(std::cin,input);
    while (std::cin && input.size() > 0)
    {
        if (isPal(input))
            std::cout << "That was a palindrome!\n";
        else
            std::cout << "That was not a palindrome!\n";
        std::cout << "Enter a string (empty string to quit):\n";
        std::getline(std::cin,input);
    }
    std::cout << "Bye!\n";

    return 0;
}

bool isPal(const std::string & s)
{
    std::string rev(s.rbegin(), s.rend()); // construct reversed
string

    // some older compilers don't implement the above constructor
    // another approach is this
    // std::string rev(s);    // rev same size as s
    // copy(s.rbegin(), s.rend(), rev.begin());

    return (rev == s);
}
```

PE 16-4

```
// pe16-4.cpp -- one possibility

#include <iostream>
#include <algorithm>
#define MAX 10

int reduce(long ar[], int n);
void show(const long ar[], int n);
```

```
int main()
{
    long myarray[MAX] = {12, 12 ,5, 6, 11, 5, 6, 77, 11,12};

    show(myarray, MAX);

    int newsize = reduce(myarray,MAX);
    show(myarray, newsize);
    return (0);
}

int reduce(long ar[], int n)
{
    // or one could copy to a list and use list methods
    // or copy to a set; in either case, copy results
    // back to array
    std::sort(ar, ar + n);
    long * past_end;
    past_end = std::unique(ar, ar + n);
    return past_end - ar;
}

void show(const long ar[], int n)
{
    for (int i = 0; i < n; i++)
        std::cout << ar[i] << ' ';
    std::cout << std::endl;
}
```

PE 16-8

```
// pe16-8.cpp

#include <iostream>
#include <set>
#include <algorithm>
#include <iterator>
#include <cstdlib>
#include <string>

int main()
```

```
{
    using namespace std;
    string temp;

    set<string> mats;
    cout << "Enter Mat's guest list (empty line to quit):\n";
    while (getline(cin,temp) && temp.size() > 0)
        mats.insert(temp);
    ostream_iterator<string,char> out (cout, "\n");
    cout << "Mat's guest list:\n";
    copy(mats.begin(), mats.end(), out);

    set<string> pats;
    cout << "Enter Pat's guest list (empty line to quit):\n";
    while (getline(cin,temp) && temp.size() > 0)
        pats.insert(temp);
    cout << "\nPat's guest list:\n";
    copy(pats.begin(), pats.end(), out);

    set<string> both;
    set_union(mats.begin(), mats.end(), pats.begin(),
pats.end(),
        insert_iterator<set<string> >(both, both.begin()));
    cout << "\nMerged guest list:\n";
    copy(both.begin(), both.end(), out);

    return 0;
}
```

Chapter 17

PE 17-1

```
// pe17-1.cpp
#include <iostream>

int main(void)
{
    using namespace std;
    char ch;
    int count = 0;
```

```
    while (cin.get(ch) && ch != '$')
        count++;
    if (ch == '$')
        cin.putback(ch);
    else
        cout << "End of input was reached\n";
    cout << count << " characters read\n";
    cin.get(ch);
    cout << "Then next input character is " << ch << endl;
    return 0;
}
```

PE 17-3

```
// pe17-3.cpp
#include <iostream>
#include <fstream>
#include <cstdlib>

int main(int argc, char * argv[])
{
    using namespace std;
    if (argc < 3)
    {
        cerr << "Usage: " << argv[0]
            << " source-file target-file\n";
        exit(EXIT_FAILURE);
    }
    ifstream fin(argv[1]);
    if (!fin)
    {
        cerr << "Can't open " << argv[1] << " for input\n";
        exit(EXIT_FAILURE);
    }
    ofstream fout(argv[2]);
    if (!fout)
    {
        cerr << "Can't open " << argv[2] << " for output\n";
        exit(EXIT_FAILURE);
    }
```

```
    char ch;
    while (fin.get(ch))
        fout << ch;
    cout << "Contents of " << argv[1] << " copied to "
         << argv[2] << endl;
    fin.close();
    fout.close();
    return 0;
}
```

PE 17-5

```
// pe17-5.cpp
#include <iostream>
#include <fstream>
#include <set>
#include <algorithm>
#include <iterator>
#include <cstdlib>
#include <string>

int main()
{
    using namespace std;
    ifstream mat("mat.dat");
    if (!mat.is_open())
    {
        cerr << "Can't open mat.dat.\n";
        exit(1);
    }
    ifstream pat("pat.dat");
    if (!pat.is_open())
    {
        cerr << "Can't open pat.dat.\n";
        exit(1);
    }

    ofstream matnpat("matnpat.dat");
    if (!matnpat.is_open())
    {
        cerr << "Can't open pat.dat.\n";
```

```
        exit(1);
    }

    string temp;

    set<string> mats;
    while (getline(mat,temp))
        mats.insert(temp);
    ostream_iterator<string,char> out (cout, "\n");
    cout << "Mat's guest list:\n";
    copy(mats.begin(), mats.end(), out);

    set<string> pats;
    while (getline(pat,temp))
        pats.insert(temp);
    cout << "\nPat's guest list:\n";
    copy(pats.begin(), pats.end(), out);

    ostream_iterator<string,char> fout (matnpat, "\n");
    set<string> both;
    set_union(mats.begin(), mats.end(), pats.begin(),
pats.end(),
        insert_iterator<set<string> >(both, both.begin()));
    cout << "\nMerged guest list:\n";
    copy(both.begin(), both.end(), out);
    copy(both.begin(), both.end(), fout);

    return 0;
}
    if (!pat.is_open())
    {
        cerr << "Can't open pat.dat.\n";
        exit(1);
    }

    ofstream matnpat("matnpat.dat");
    if (!matnpat.is_open())
    {
        cerr << "Can't open pat.dat.\n";
        exit(1);
    }

    string temp;
```

```
    set<string> mats;
    while (getline(mat,temp))
        mats.insert(temp);
    ostream_iterator<string,char> out (cout, "\n");
    cout << "Mat's guest list:\n";
    copy(mats.begin(), mats.end(), out);

    set<string> pats;
    while (getline(pat,temp))
        pats.insert(temp);
    cout << "\nPat's guest list:\n";
    copy(pats.begin(), pats.end(), out);

    ostream_iterator<string,char> fout (matnpat, "\n");
    set<string> both;
    set_union(mats.begin(), mats.end(), pats.begin(),
pats.end(),
        insert_iterator<set<string> >(both, both.begin()));
    cout << "\nMerged guest list:\n";
    copy(both.begin(), both.end(), out);
    copy(both.begin(), both.end(), fout);

    return 0;

}
```

PE 17-7

```
// pe17-7.cpp
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <algorithm>
#include <cstdlib>

void ShowStr(const std::string & s);
void GetStrs(std::istream & is, std::vector<std::string> & vs);

class Store
{
```

```
public:
    std::ostream & os;
    Store (std::ostream & o) : os(o) {}
    void operator()(const std::string &s);
};

int main()
{
    using namespace std;
    vector<string> vostr;
    string temp;

// acquire strings
    cout << "Enter strings (empty line to quit):\n";
    while (getline(cin,temp) && temp[0] != '\0')
        vostr.push_back(temp);
    cout << "Here is your input.\n";
    for_each(vostr.begin(), vostr.end(), ShowStr);

// store in a file
    ofstream fout("strings.dat", ios_base::out |
ios_base::binary);
    for_each(vostr.begin(), vostr.end(), Store(fout));
    fout.close();

// recover file contents
    vector<string> vistr;
    ifstream fin("strings.dat", ios_base::in | ios_base::binary);
    if (!fin.is_open())
    {
        cerr << "Could not open file for input.\n";
        exit(EXIT_FAILURE);
    }
    GetStrs(fin, vistr);
    cout << "\nHere are the strings read from the file:\n";
    for_each(vistr.begin(), vistr.end(), ShowStr);

    return 0;
}

void ShowStr(const std::string & s)
{
    std::cout << s << std::endl;
}
```

```
void Store::operator()(const std::string &s)
{
    std::size_t len = s.size();
    os.write((char *)&len, sizeof(std::size_t));
    os.write(s.data(), len);
}

void GetStrs(std::istream & is, std::vector<std::string> & vs)
{
    std::string temp;
    size_t len;

    while (is.read((char *) &len, sizeof(size_t)) && len > 0)
    {
        char ch;
        temp = "";
        for (int j = 0; j < len; j++)
        {
            if (is.read(&ch, 1))
            {
                temp += ch;
            }
            else
                break;
        }
        if (is)
            vs.push_back(temp);
    }
}
```