

课后答案网 您最真诚的朋友



www.hackshp.cn网团队竭诚为学生服务，免费提供各门课后答案，不用积分，甚至不用注册，旨在为广大学生提供自主学习的平台！

课后答案网：www.hackshp.cn

视频教程网：www.efanjy.com

PPT课件网：www.ppthouse.com

课后答案网
www.hackshp.cn

第1章习题答案

1. 答：C++语言是支持面向对象程序设计的最主要的代表语言之一：封装和信息隐藏；抽象数据类型；继承和派生的方式实现程序的重用机制；多态特征；模板技术。

C++语言是 C 语言的超集

C++与 C 语言相比具有以下特点：

- 从 C 语言中继承了其独有的为程序员所喜爱的简明高效的表达式形式。
- 比较容易地解决了目标代码高质量、高效率的问题。
- 吸引了 20 世纪 80 年代成长起来的一批高水平的 C 程序员，他们比较自然地转向 C++。
- 可以与 80 年代以来的大批 C 程序软件兼容，可以使它们在 C++环境下继续维护使用。
-

2. 答：将复杂的过程简单地按功能分层从而达到解决问题的目的，这种思想就是过程化程序设计的思想。过程化程序设计以一系列过程的划分和组织来观察、分析和解决问题。

3. 答：在编程方法的改进中，人们首先适应了用抽象数据类型来描述数据结构。这种编程方法是基于对象的程序设计。对象是程序中抽象数据类型的具体表现。编程模式变成了：程序=算法+抽象数据类型。

基于对象的程序设计本质上还是过程化的。类的层次关系也带来了对象的层次关系，它反映了同种操作的异类行为。对象之间一种和谐的共存关系，而不是你命令我，我强迫你的关系。有什么任务，通过接口由抽象数据类型定义）请求。程序的运行，是对象通过抽象数据类型来表现和沟通，而抽象数据类型最后还是以对象来实例化的，程序的形式变成了：程序=（对象，对象，对象，……）。

4. 答：编辑、编译、连接和运行

5. 答：

```
#include<iostream.h>
void main ()
{
    cout<<"I am a student."<<endl;
}
```

6. 答：

```
#include<iostream.h>
void main ()
{
    cout<<"    "<<endl;
    cout<<"   ***"<<endl;
    cout<<"  *****"<<endl;
    cout<<"*****"<<endl;
    cout<<"   ***"<<endl;
    cout<<"    "<<endl;
    cout<<"    "<<endl;
}
```

第 2 章习题答案

一、选择题

1. b 2. a 3. b 4. c 5. d 6. b 7. a 8. d 9. c 10. b

二、填空题

1. 1 2. 12.5 3. $x \leq 2$ 4. 5. 右 左
6. $x*(y+8)$ 7. 27 8. 14 9. 35 10. 103

三、编程及简答题

1. A1=10 A2=0
2. 算数运算符由“+，-，*，/，%”构成。
3. 由高到低分别是：‘*’、‘+’、‘>’、‘>=’、‘&’、‘&&’、‘*=’。
4. ‘-’可以代表减号和负号；‘&’可以代表取地址和按位与；‘*’可以代表乘号和取内容
5. ‘+’由左至右；‘&（位）’由左至右；‘=’由右至左；‘||’由左至右；
6. (1) 3 (2) 6 (3) 4.5 (4) 444 (5) 188 (6) 238

7.

```
#include <iostream.h>
```

```
void main()
```

```
{
```

```
    double a,b;
```

```
    cin>>a>>b;
```

```
    if(a>b)
```

```
        cout<<"max is"<<a<<endl;
```

```
    else
```

```
        cout<<"max is"<<b<<endl;
```

```
}
```

8. (1)a=5 b=0 (2) a=6 b=0

9. a=5 b=10

10.

```
#include <iostream.h>
```

```
void main()
```

```
{
```

```
    cout<<sizeof(int)<<endl;
```

```
    cout<<sizeof(float)<<endl;
```

```
    cout<<sizeof(double)<<endl;
```

```
    cout<<sizeof(long)<<endl;
```

```
    cout<<sizeof(char)<<endl;
```

```
}
```

11.

```
#include <iostream.h>
```

```
void main()
```

```
{
```

```

int a=42486;
cout<<oct<<a<<endl;
cout<<hex<<a<<endl;
unsigned short b=42486;
cout<<dec<<b<<endl;
}

```

12.

```

#include <iostream.h>
#include <iomanip.h>
void main()
{
    double pi=3.1415926;
    double r1=40, r2=928.335;
    cout<<setw(13)<<pi<<endl;
    cout<<setw(13)<<r1<<endl;
    cout<<setw(13)<<pi*r1*r1<<endl;
    cout<<setw(13)<<r2<<endl;
    cout<<setw(13)<<pi*r2*r2<<endl;
}

```

13.

```

#include <iostream.h>
#include <iomanip.h>
void main()
{
    const double e=2.718281828;
    cout<<setw(10)<<e<<endl;
    cout<<setprecision(8)<<e<<endl;
}

```

14.

```

#include <iostream.h>
#include <iomanip.h>
void main()
{
    const double c=500;
    cout<<"\How many students here?"<<endl;
    cout<<"\ "<<c<<"\ "<<endl;
}

```

15.

```

#include <iostream.h>
void main()
{
    cout<<"size of int "<<sizeof(int)<<" byte"<<endl;
    cout<<"size of short "<<sizeof(short)<<" byte"<<endl;
    cout<<"size of long "<<sizeof(long)<<" byte"<<endl;
}

```

```
cout<<"size of unsigned int "<<sizeof(unsigned int)<<" byte"<<endl;
cout<<"size of unsigned short  "<<sizeof(unsigned short )<<" byte"<<endl;
cout<<"size of unsigned long  "<<sizeof(unsigned long )<<" byte"<<endl;
}
```

16.

(1) a= -107374176.00,b= -107374176.00, c= -107374176.00 narea of triangle is 0.5f

(2) 输入三角形的三个边， 计算面积

(3)

```
#include<stdio.h>
#include<iostream.h>
#include<math.h>
void main ()
{
float a,b, c, s, area;
cout<<"Input 3 sides of one triangle:\n";
cin>>a>>b>>c;
s= (a + b+ c)/2;
area= sqrt(s* (s-a) * (s-b) * (s- c));
cout<<"a="<<a<<"b="<<b<<"c="<<c<<endl;
cout<<"area of triangle is %l0.5f"<<area;
}
```

(4)

```
float getarea(float a,float b,float c)
{
float s= (a + b+ c)/2;
return sqrt(s* (s-a) * (s-b) * (s- c));
}
```

17.

In main():

Enter two numbers:

2

10

Calling add():

In add(),received2 and10

and return12

Back in main():

cwas set to12

18.

```
#include<iostream.h>
double getv(double r,double h)
{
return 3.14159*r*r*h;
```

```

}
void main ()
{
double r,h;
cin>>r>>h;
cout<<getv(r,h)<<endl;
}

```

第 3 章习题答案

一、选择题

1. d 2. d 3. a 4. d 5. d 6. b

二、填空题

1. do-while 先执行一次循环体，然后判断是否继续执行循环体，while 语句先判断，若真则执行循环体。
2. x 的值为 10，y 的值为 20，z 的值为 30。
3. break 语句实现的功能是终止本次循环，跳出循环体。
4. continue 语句实现的功能是终止本次循环，接着执行下次循环。

三、编程及简答题

1. 参考代码如下：

```

for(int i=2; i<=20; i++)
{
    bool b=false;
    for(int j=2; j<i; j++)
    {
        if(i%j == 0)
        {
            b=true;
            break;
        }
    }
    if(!b)
        cout<<i<<"是素数"<<endl;
}

```

2. 参考代码如下：

```

#include <iostream.h>
void main()
{
    char ch;
    cin>>ch;
    if(ch>='0' && ch<='9')
        cout<<"数字"<<endl;
    else if((ch>='a' && ch<='z')||(ch>='A' && ch<='Z'))

```

```

        cout<<"字母"<<endl;
    else
        cout<<"特殊字符"<<endl;
}

```

3. 参考代码如下:

```
#include<iostream.h>
```

```
void main()
```

```
{
```

```
    char c;
```

```
    while(true)
```

```
    {
```

```
        cout << "MAIN MENU:" << endl;
```

```
        cout << "l: left, r: right, q: quit -> ";
```

```
        cin >> c;
```

```
        switch(c)
```

```
        {
```

```
        case 'q':
```

```
            break;
```

```
        case 'l':
```

```
            cout << "LEFT MENU:" << endl;
```

```
            cout << "select c or d: ";
```

```
            cin >> c;
```

```
            if(c == 'c')
```

```
            {
```

```
                cout << "you chose 'c'" << endl;
```

```
                continue; // 返回主菜单
```

```
            }
```

```
            if(c == 'd')
```

```
            {
```

```
                cout << "you chose 'd'" << endl;
```

```
                continue; // 返回主菜单
```

```
            }
```

```
        else
```

```
        {
```

```
            cout << "you didn't choose c or d!"<< endl;
```

```
            continue; //返回主菜单
```

```
        }
```

```
        break;
```

```
    case 'r':
```

```
        cout << "RIGHT MENU:" << endl;
```

```
        cout << "select a or b: ";
```

```
        cin >> c;
```

```
        if (c == 'a')
```

```
        {
```

```

        cout << "you chose 'a'" << endl;
        continue; //返回主菜单
    }
    if (c == 'b')
    {
        cout << "you chose 'b'" << endl;
        continue; //返回主菜单
    }
    else
    {
        cout << "you didn't choose c or d!" << endl;
        continue; //返回主菜单
    }
    break;
}
cout << "you must type l or r or q!" << endl;
}
cout << "quitting menu..." << endl;
}

```

4. 此程序的功能是判断输入是否为'b'，是则输出响铃，否则输出“bye!”。

5. 参考代码如下：

```
#include <iostream.h>
```

```
void main()
```

```

{
    int a,b,c,temp;
    cin>>a>>b>>c;
    if(a>=b)
    {
        if(a>=c)
            cout<<"最大的数是:"<<a<<endl;
        else
            cout<<"最大的数是:"<<b<<endl;
    }
    else
    {
        if(b>=c)
            cout<<"最大的数是:"<<b<<endl;
        else
            cout<<"最大的数是:"<<c<<endl;
    }
}
}

```

6. (1) else 与 if (a<0)配对。(2) else 与 if(a==0)配对。

7. 看下面几个循环语句，说明循环的含义。

(1)、判断输入的字符是否为'Y'，'y'，'N'，'n'其中的一个。

(2)、无限次循环。

(3)、判断 i 的值是否为非 0。

(4)、程序求出 1 到 100 之间的正整数的和。While 判断循环是否超出 100。

8. 参考代码如下：

```
#include <iostream.h>
void main()
{
    char ch[100]={0};
    cin>>ch;
    int i=0,a=0,b=0,c=0;
    while(ch[i]!='\0')
    {
        if(ch[i]>='0' && ch[i]<='9')
            a++;
        else if((ch[i]>='a' && ch[i]<='z')||(ch[i]>='A' && ch[i]<='Z'))
            b++;
        else
            c++;
        i++;
    }
    cout<<"数字个数:"<<a<<endl
        <<"字母个数:"<<b<<endl
        <<"其他符号个数:"<<c<<endl;
}
```

9. 参考代码如下：

```
#include <iostream.h>
void main()
{
    int limit;
    long n;
    double sum=0;
    cin>>limit;
    for(n=1;;n++)
    {
        sum=sum+1.0/n;
        if(sum>limit)
        {
            cout<<n<<endl;
            break;
        }
    }
}
```

10. 参考代码如下：

```
#include <iostream.h>
```

```

void main()
{
    int f1=1,f2=1,f3;
    cout<<f1<<endl;
    cout<<f2<<endl;
    for(int i=3;i<=20;i++)
    {
        f3=f1+f2;
        cout<<f3<<endl;
        f1=f2;
        f2=f3;
    }
}

```

11. 参考代码如下:

```
#include <iostream.h>
```

```
void main()
```

```
{
```

```
int a=0;
```

```
cout<<"输入学生成绩:";
```

```
cin>>a;
```

```
cout<<"输入学生成绩为:";
```

```
switch(a/10)
```

```
{
```

```
case 0:
```

```
case 1:
```

```
case 2:
```

```
case 3:
```

```
case 4:
```

```
case 5:
```

```
    cout<<"不及格!"<<endl;
```

```
    break;
```

```
case 6:
```

```
    cout<<"及格!"<<endl;
```

```
    break;
```

```
case 7:
```

```
    cout<<"中!"<<endl;
```

```
    break;
```

```
case 8:
```

```
    cout<<"良!"<<endl;
```

```
    break;
```

```
case 9:
```

```
case 10:
```

```
    cout<<"优秀!"<<endl;
```

```
    break;
```

```
}  
}
```

第 4 章习题答案

1. 答：数组是同类型数据的有序集合，集合中的每一个元素称为数组元素。数组在存储时将占用连续的内存空间，其大小是数组中元素个数与数组元素类型所占内存空间大小的乘积。

2. 数组元素可以未经赋值而参与其他运算吗？为什么？

答：不能，创建数组后，各元素的值是由系统随即分配的，若没有赋值参与运算后得到的值是不可预料的。

3. 答：指针是一个用来指示一个内存地址的变量。定义指针没有赋予初始值时，系统会随机为这个指针分配一个值，使其只相一个存储空间，若直接引用时，可能存在非法访问，若将指针赋值为空时，引用不会非法。

4. 答：*运算符是取出指针指向存储空间的值，&运算符是取出变量的地址

5. 答：p++运算是将 p 指向下一个存储单元，p-2 代表取出 p 之前的两个存储单元的地址。

6. 答：指针数组是个数组，数组的元素是指针。数组指针是指针，指向某一个数组，其声明方法与一般指针相同，赋值时接收某一数组的首地址。

7. 答：指针常量是指该指针的指向不可被更新，常量指针是指该指针指向的变量不可被更新。

8. 答：数组一旦定义，它的空间就是固定的，可能造成预估不足而溢出或预留过量造成浪费，动态内存分配根据需要动态的申请，撤销。C 语言中可以使用函数 malloc, realloc, free, C++中除了可以使用 C 提供的方法外，还可以使用 new 和 delete 运算符实现。

9. 参考代码如下：

```
struct student  
{  
    char num[10];  
    char name[10];  
    char sex[2];  
    int age;  
};  
struct student stu1;  
stu1.age=20;  
strcpy(stu1.name,"张三");  
cout<<"stu1 的姓名是:"<<stu1.name<<endl;
```

10. 参考代码如下：

```
struct Class  
{  
    char name[20];  
    int studentcout;  
    struct student stu[40];  
};
```

```
struct Class cls1;
strcpy(cls1.name,"08 计算机网络 1 班");
cout<<"班级名称是:"<<cls1.name<<endl;
```

11. typedef 是对已有的数据类型的重新起个别名，它不可以创建新类型？

12. 答：可以使用枚举类型，参考代码如下：

```
enum position{SMY,QF,ZF,HW};
```

13. 答：引用就是某一变量（目标）的一个别名，对引用的操作与对变量直接操作完全一样。使用引用应该注意：声明一个引用，不是新定义了一个变量，它只表示该引用名是目标变量名的一个别名，它本身不是一种数据类型，因此引用本身不占存储单元，系统也不给引用分配存储单元；不能建立数组的引用；引用声明完毕后，相当于目标变量名有两个名称，即该目标原名称和引用名，且不能再把该引用名作为其他变量名的别名；声明引用时，必须同时对其进行初始化。

14. 参考代码如下：

```
#include <iostream.h>
```

```
void main()
```

```
{
    int t=0,i,j,a[5][5],b[5][5];
    for(i=0;i<5;i++)
    {
        for(j=0;j<5;j++)
        {
            cin>>a[i][j];
        }
    }
    for(i=0;i<5;i++)
    {
        for(j=0;j<5;j++)
        {
            t=a[i][j];
            a[i][j]=a[j][i];
            a[j][i]=t;
        }
    }
}
```

15. 参考代码如下：

```
#include <iostream.h>
```

```
void main()
```

```
{
    char ch[100];
    int i=0;
    cin.getline(ch,100);
    while(1)
    {
        while(ch[i]!='&&ch[i]!='\0')

```

```

    {
        cout<<ch[i];
        i++;
    }
    cout<<endl;
    if(ch[i]=='\0')
        break;
    i++;
}
}

```

16. (1) 运行结果:

A[0]=-1 B[0]=17

A[1]=1 B[1]=15

A[2]=3 B[2]=13

A[3]=5 B[3]=11

A[4]=7 B[4]=9

A[5]=9 B[5]=7

A[6]=11 B[6]=5

A[7]=13 B[7]=3

A[8]=15 B[8]=1

A[9]=17 B[9]=-1

(2) 运行结果:

Output int i=10

Output int pointer i=10

(3) 运行结果:

The weight of dog1 is 15

The weight of dog2 is 37

The weight of chicken is 3

(4) 运行结果:

The game was played and we won!

The game was played and we lost.

The game was played

The game was cancelled

第 5 章习题答案

1. 答: 函数是某特定功能的抽象。发生函数调用时, 首先由系统进行现场保护, 才转去执行函数本身的代码, 函数结束或遇到 `return` 语句时从系统获得返回地址, 转回原位置继续程序的执行。

2. 答: 形式参数是函数声明或定义时在参数列表中给出的参数, 它必须有类型的声明, 不能为常量, 并且函数被调用前不被创建; 实际参数是函数被调用时在参数列表中给出的参数,

它可以是任何有确切值的表达式（包括常量、有返回值的函数调用），函数调用时由实际参数完成形式参数的初始化。值传递是单向的，即实际参数完成形式参数的初始化赋值后即完成使命，形参在函数体内所做修改均不影响实参；址传递是双向的，即实际参数除完成形式参数的初始化赋值外，形参在函数体内所做修改均直接体现在实参上。

3. 答：函数的嵌套调用是指函数执行中又调用其他函数。

函数的递归调用是指函数执行中直接或间接地调用其自身。

4. 答：函数可以先有原型声明即可被调用，函数的定义在调用之后，这样有利于保持程序结构的清晰以及外部函数在多文件组织时的共享。原型声明的格式为：

函数类型 函数名（类型 形参 1，类型 形参 2...）；

形参名此时可以省略。

5. 答：默认形参值的给定必须按参数表由后向前不间断的给出，并且当有函数重载时，应避免默认值所造成的歧意。

6. 答：内联函数的声明格式为：**inline** 函数类型 函数名（形参表）；

内联函数是为节省函数调用时的转移开销而设，因此内联函数一般为调用频繁而执行简单的函数，不能有循环语句和 **switch** 语句，也不能进行异常接口声明。

7. 答：函数重载可以提高代码的抽象度和可读性。重载函数是以参数表的参数个数或类型加以区分的。

8. 参考代码：

```
#include "iostream.h"
float fun(float x)
{
    return (x-32)*5/9;
}
void main()
{
    float f,c;
    cout<<"请输入华氏温度： ";
    cin>>f;
    c=fun(f);
    cout<<"转换后摄氏温度为： "<<c<<endl;
}
```

运行结果：

请输入华氏温度： 104

转换后摄氏温度为： 40

9. 参考代码：

```
#include "iostream.h"
#define MAXSIZE 50
void count(char *s,int &c,int &n)
{
    c=n=0;
    int i=0;
    while(s[i])
    {
        if(s[i]>='a'&& s[i]<='z'||s[i]>='A'&& s[i]<='Z')
```

```

        c++;
        else if(s[i]>='0'&& s[i]<='9')
            n++;
        i++;
    }
}

void main()
{
    char text[MAXSIZE];
    int x,y;
    cout<<"请输入任意字符: "<<endl;
    cin.getline(text,MAXSIZE,'\n');//有效长度不超过 MAXSIZE-1
    count(text,x,y);
    cout<<"英文字母个数: "<<x<<endl;
    cout<<"阿拉伯数字个数: "<<y<<endl;
}

```

运行结果:

请输入任意字符:

this is a test.1+1=2.end.

英文字母个数: 14

阿拉伯数字个数: 3

10. 参考代码: 分析: 得公式 $\sum_{i=1}^n (-1)^{i+1} \frac{1}{i}$

```
#include "iostream.h"
```

```
int power(int p,int q)
```

```
{
    int result=1;
    for(int s=0;s<q;s++)
        result*=p;
    return result;
}
```

```
double sum(int n)
```

```
{
    double result=0;
    for(int k=1;k<=n;k++)
        result+=(double)power(-1,k+1)/k;
    return result;
}
```

```
void main()
```

```
{
    int n;
    cin>>n;
    cout<<sum(n)<<endl;
}
```

```
}
```

运行结果:

5

0.783333

11. 参考代码:

```
#include "iostream.h"
```

```
int code(int n)
```

```
{
```

```
    if(n==10)
```

```
        return 110;
```

```
    return 2*code(n+1)-10;
```

```
}
```

```
void main()
```

```
{
```

```
    cout<<"第 1 个人的密码是: "<<code(1)<<endl;
```

```
}
```

运行结果:

第 1 个人的密码是: 51210

12. 参考代码:

```
#include "iostream.h"
```

```
double vol(double x)
```

```
{
```

```
    cout<<"正方体体积计算..."<<endl;
```

```
    return x*x*x;
```

```
}
```

```
double vol(double r,double pi)
```

```
{
```

```
    cout<<"球体积计算..."<<endl;
```

```
    return pi*r*r*r*3/4;
```

```
}
```

```
double vol(double r,double height,double pi)
```

```
{
```

```
    cout<<"圆锥体积计算..."<<endl;
```

```
    return pi*r*r*height/3;
```

```
}
```

```
void main()
```

```
{
```

```
    cout<<vol(10)<<endl;
```

```
    cout<<vol(10,3.14)<<endl;
```

```
    cout<<vol(10,10,3.14)<<endl;
```

```
}
```

运行结果:

正方体体积计算...

1000

球体积计算...

2355

圆锥体积计算...

1046.67

第 6 章习题答案

1. 答：标识符的作用域指的是它在程序文件中的有效范围。C++程序中标识符的作用域可分为局部作用域和全局作用域 2 种。(1) 局部作用域①函数原型作用域——在进行函数的原型声明时，形参列表中使用的标识符则具有函数原型作用域，它的有效范围仅限于形参列表的一对圆括号内。②块作用域——在某一个函数的完整定义中或是复合语句中说明的标识符，其作用域是块作用域，它的有效范围是从该标识符被说明处开始，到函数定义完成或复合语句的结束。(2) 全局作用域——没有被任何定界符限定起来的标识符，具有全局作用域，有效范围起始于该标识符被说明的地方，终止于整个文件结束。

标识符的可见性是指标识符在作用域内是否可以被使用。标识符命名时要求相同的作用域内不允许标识符重名，但是在不同的作用域内可以重名。一般情况下，标识符在各自的作用域内均可见，但当标识符重名，且作用域发生重叠时，要注意其可见性的应用规则为小作用域优先。

2. 答：生存期有静态生存期和动态生存期两种。静态生存期的变量从编译时即被创建并保留其内存空间直到整个程序运行结束；动态生存期变量则在运行时，因其定义语句的执行才被创建，当其作用域结束时内存空间即被收回。

3. 答：从作用域上看，全局变量具有全局作用域，局部变量和静态局部变量具有局部作用域；从生存期上看，全局变量和静态局部变量具有静态生存期，局部变量具有动态生存期。

4. 参考代码：

```
#include "iostream.h"
```

```
int fac(int i)
```

```
{
```

```
    static int x=1;
```

```
    return x*=i;
```

```
}
```

```
void main()
```

```
{
```

```
    int n;
```

```
    cin>>n;
```

```
    for(int i=1;i<=n;i++)
```

```
    {
```

```
        cout<<i<<"!="<<fac(i)<<endl;
```

```
    }
```

```
}
```

运行结果：

10

1!=1

2!=2
3!=6
4!=24
5!=120
6!=720
7!=5040
8!=40320
9!=362880
10!=3628800

5. 答:

所谓命名空间，是一种将程序库名称封装起来的方法。其格式为：

namespace 命名空间名称

{

//本空间下各种变量、函数的声明或定义

}

它就像在各个程序库中立起一道道围墙，不同空间下的标识符不会冲突。

6. 答:

文件包含预处理命令的其格式为:

#include<文件名>

或#include"文件名"

其含义是：在编译前，用命令中的文件名所指定的文件内容替换该命令。便于实现多文件组织，提高代码利用率及编程效率。

7. 答:

在进行带参数宏的定义时要注意，为确保参数在宏替换的过程中不产生解释上的错误，要将<文字串>中的参数用圆括号括起来。

8. 答:

条件编译与宏定义相结合，可以实现依不同条件选择某程序文件（或该文件的一部分）是否参与多文件组织结构。

9. 答：运行结果：

222

323

222

10. 答:

错误：变量 x 重复定义

改正方法 1：去掉 1.h 中 x 的初始化赋值。

改正方法 2：去掉 main.cpp 中 x 的定义。

附加题：

//-----Circle.h-----

#ifndef _CIRCLE_H

#define _CIRCLE_H

class Circle

{

private:

int r;

```

public:
    Circle(int xr);
    ~Circle();
    double Area();
};
#endif
//-----Circle.cpp-----
#include " Circle.h"
Circle::Circle(int xr)
{
    r=xr;
}
Circle::~~Circle()
{}
double Circle::Area()
{
    return 3.14*r*r;
}

//-----Ring.h-----
////此处的条件编译可省略
////#ifndef _RING_H
////#define _RING_H
#include "2.h"
class Ring
{
private:
    Circle ro,ri;
public:
    Ring(int xro,int xri);
    ~Ring();
    double Area();

};
////#endif
//-----Ring.cpp-----
#include " Ring.h"
Ring::Ring(int xro,int xri):ro(xro),ri(xri)
{}
Ring::~~Ring()
{}
double Ring::Area()
{
    return ro.Area()-ri.Area();
}

```

```
//-----main.cpp-----
#include "iostream.h"
#include "Circle.h"
#include "Ring.h"
void main()
{
    Circle c(5);
    Ring r(10,5);
    cout<<c.Area()<<endl;
    cout<<r.Area()<<endl;
}
```

第 7 章习题答案

1.答：①从现实世界中客观存在的事物（即对象）出发来构造软件系统，并在系统构造中尽可能运用人类的自然思维方式。②从问题域中客观存在的事物出发来构造软件系统，用对象作为对这些事物的抽象表示，并以此为系统的基本构成单位；事物的静态特征用对象的属性表示；对象的属性和方法合为一体，成为一个独立的实体，对外不公开其内部特征和实现细节；对所有事物进行分类，把具有相同属性和方法的对象归为一类，类是这些对象的抽象描述；复杂对象可以由简单对象作为其构成部分；对象之间通过消息进行通信，实现对象之间的动态联系；通过关联表达对象之间的静态关系。

2.答：①现实世界中的任何事物都可以称为对象，它是大量的、无处不在的。世界就是一个对象集合。②一个对象由一组属性和对这组属性进行操作的方法构成。属性是用来描述对象静态特征的一个数据项，方法是指由对象完成的任务。例如：冰箱是一个对象，它的温度是多少、冰箱的位置、冰箱门是否打开等，这些都是冰箱特有的属性，打开冰箱门、闭合冰箱门是它的方法。

3.答：①分类的方法和原则是抽象，忽略事物的非本质特征，只注意那些与当前目标有关的本质特征，从而找出事物的共性，把具有共同性质的事物划分为一类，得到一个抽象的概念，这个概念就是类。②类与对象的关系就如同模具与用模具制造出来的铸件的关系，他们之间是抽象和具体的关系。类是对多个对象进行综合抽象的结果，对象又是类的个体实物，一个对象是类的一个实例。

4.答：①类有以下特点：类是一个定义属性和方法的模版；类由程序员编写，并作为程序的一部分；程序执行时，类本身并不存在，而是以一个或多个成员对象的形式出现；在程序执行期间，类的代码不能修改，从这个意义上讲，类是静态的；类由类名指定。②对象则有以下特点：对象必须属于某个类；对象在程序执行期间存在；对象必须显式声明，而且要由执行程序来构造；对象具有属性和方法，在程序执行期间可以修改其属性值，也可以执行方法；对象通常用一个变量名来引用。

5.答：①消息一个对象要求另一个对象执行某个操作的规格说明，通过消息传递才能完成对象之间的相互请求和协作。②同一对象可以接受不同形式的多个消息并作出不同的响应；相同形式的消息可以传递给不同的对象，作出的响应可以是不同的；消息的发送可以不考虑具体的接受者，对象可以响应，也可以不响应。

6.答：①面向对象程序设计的核心思想是数据的分解，着重点放在被操作的数据上而不是实

现操作的过程上。它把数据及其操作作为一个整体对待，数据本身不能被外部过程直接存取。

②面向对象系统的三个基本特征是：封装、继承、多态。

7 答：①它可以使用现有类的所有功能，并在无需重新编写原来的类的情况下对这些功能进行扩展。②通过继承创建的新类称为“子类”或“派生类”。被继承的类称为“基类”、“父类”或“超类”。比如：狗具有哺乳动物的所有特性，同时和具有区别于其他哺乳动物（如猫，大象）的特征。

8 答：①将对象的属性和行为放在一起作为一个整体的方法称为封装，它将对象的大部分行为的实现隐蔽起来，仅通过一个可控的接口与外界交互。②数据的封装隐藏了抽象的内部实现细节。封装是将数据抽象的外部接口与内部的实现细节清楚地分离开，使人们只关心如何操作，而不必在意实现细节。例如：录音机为例，录音机上有若干按键，当人们使用录音机时，只要根据自己的需要，如放音，录音，停止，倒带等，按下与之对应的键，录音机就会完成相应的工作。这些按键安装在录音机的表面，人们通过它们与录音机交互。我们无法（当然也没有必要）操作录音机的内部电路，因为他们被装在机壳里，录音机的内部情况对于用户来说是隐蔽的，不可见的。

9 答：①在面向对象思想中，多态是指同一个方法名（具有相同的操作）可作用于不同的对象，从而执行不同的方法代码并获得不同的结果，这种机制就称为多态。②就是为了类在继承和派生的时候，保证使用“家谱”中任一类的实例的某一属性时的正确调用。

10 答：①C++语言支持两种多态②动态多态和静态多态，分别是通过两种联编方式来实现的（动态联编和静态联编）。通常的方法调用使用的是静态联编，因为编译时就已经明确知道要使用哪一个方法。在动态联编中，编译器为方法调用生成一段代码，程序运行时，这段代码根据对象的类型来计算究竟应该使用哪一个方法。

www.hackshp.cn

第 8 章习题答案

1. 答：有三种，分别是保护，公有和私有。公有成员在类内或类外均可被调用，保护成员只能在类内或子类内被调用，私有成员只能在类内被调用。
2. 答：两者访问权限的意义是一致的，只是成员的默认访问类型不一样，类内成员的默认访问权限是私有的，而结构体内的成员的默认权限是公有的。
3. 答：构造函数复杂创建对象，分配对象的存储空间，初始化对象中各种数据成员，在创建对象时，构造函数会被自动调用。
4. 答：在对象的生命期结束时会自动调用析构函数来释放对象所占用的存储空间。
5. 答：使用友元函数可以访问私有成员或保护成员，即在友元函数函数体内，对象可以操作私有成员或者保护成员。
6. 答：拷贝构造函数在三种情况下会被调用：
 - （1）用类的对象去初始化该类的另一个对象时。
 - （2）函数的形参是类的对象，调用函数进行形参和实参的结合时。
 - （3）函数的返回值是类的对象，函数执行完返回调用者时。
7. 答：调用拷贝构造函数是指创建一个新对象，然后在拷贝构造函数的函数体对为新创建的对象初始化数据成员，使用直接进行对象间的赋值是指，两个已经存在的对象间的数据成员的等值拷贝。
8. 答：定义类的成员时，在成员的类型前使用 `static` 进行修饰。静态数据成员可以直接由类来引用，它不依附于单个对象。静态成员函数可以直接引用该类的静态数据成员和成

员函数，不能直接引用非静态数据成员，如果要引用，必须通过参数传递的方式得到对象名，再通过对象名来引用。

9. 答：类中的成员，除了基本的数据类型定义的变量外，还可以是一个已定义的类的对象，称此做法为类的组合或类的聚合，这样的类简称为组合类。在组合类的构造函数的初始化列表里要指定子对象的构造形式。构造顺序是先按照生命顺序构造子对象，然后调用组合类的构造函数。

10. 答：参考代码如下：

```
#include <iostream.h>
class Test
{
public:
    int ma;
    void fa();
private:
    int mb;
    void fb();
protected:
    int mc;
    void fc();
    friend void f(Test &o)
    {
        cout<<o.mb;
    }
};
void main()
{
    Test a;
    a.ma++;
    a.mb++;
    f(a);
}
```

11. 答：参考代码如下：

```
#include <iostream.h>
class Date
{
    int y,m,d;
public:
    Date(int a,int b,int c)
    {
        y=a;
        m=b;
        d=c;
        cout<<"构造对象"<<endl;
    }
}
```

```

~Date()
{
    cout<<"释放对象"<<endl;
}
void Show()
{
    cout<<"年: "<<y<<" 月: "<<m<<" 日: "<<d<<endl;
}
};
void main()
{
    Date d1(2008,1,1);
    d1.Show();
}

```

12. 答: 参考代码如下:

```

#include <iostream.h>
class A
{
public:
    A()
    {
        cout<<"构造 A 对象"<<endl;
    }
};
class B
{
    A ma;
public:
    B()
    {
        cout<<"构造 B 对象"<<endl;
    }
};
void main()
{
    B o[2];
}

```

13. 答: 参考代码如下:

```

#include <iostream.h>
class A
{
    static int count;
public:
    A()

```

```

    {
        count++;
    }
    ~A()
    {
        count--;
    }
    void Show()
    {
        cout<<"目前个数为:"<<count<<"个"<<endl;
    }
};
int A::count=0;
void main()
{
    A o;
    o.Show();
}

```

14. 答：参考代码如下：

```

#include <iostream.h>
class Clock
{
    int Hour,Minute,Second;
public:
    void SetTime(int a,int b,int c)
    {
        Hour=a;
        Minute=b;
        Second=c;
    }
    void ShowTime()
    {
        cout<<"当前时间:"<<Hour<<"点"<<Minute<<"分"<<Second<<"秒"<<endl;
    }
};
void main()
{
    Clock o;
    o.SetTime(18,28,38);
    o.ShowTime();
}

```

15. 答：参考代码如下：

```

#include <iostream.h>
class A

```



```

{
public:
    A()
    {
        cout<<"构造 A 对象"<<endl;
    }
    ~A()
    {
        cout<<"释放 A 对象"<<endl;
    }
};
void main()
{
    A *p=new A;
    delete p;
}

```

16. 答：参考代码如下：

```

#include <iostream.h>
class A
{
public:
    A()
    {
        cout<<"构造"<<endl;
    }
    A(A &r)
    {
        cout<<"拷贝构造"<<endl;
    }
};
void f1(A a)
{
    ;
}
A f2()
{
    A a;
    return a;
}
void main()
{
    A o;
    f1(o);
    f2();
}

```

}

17. 答：程序运行结果：

(1)

1

2

3

2

1

0

(2)

A 类对象被构造

B 类对象被构造

A 类对象被构造

B 类对象被构造

C 类对象被构造

C 类对象被析构

B 类对象被析构

A 类对象被析构

B 类对象被析构

A 类对象被析构

18. 答：

(1)

t1.m_y++;//错误

t1.m_z=0;//错误

int b = t1.GetY();//错误;

(2)

void CTest()//构造函数不能声明有返回值

{

.....

}

19. 答：

CMyCls cls1; //调用第一个构造函数

CMyCls cls2(1); //调用第二个构造函数

CMyCls cls3(1,2); //调用第三个构造函数

20. 答：程序运行结果：

20

10

第 9 章习题答案

1. 答：A 称为 B 的子类，B 称为 A 父类。

2. 答：多继承。
3. 答：三种继承方式；继承后的成员在子类中的访问权限如下表所示。

基类中访问属性	public 继承	protected 继承	private 继承
public	public	protected	private
protected	protected	protected	private
private	不可访问	不可访问	不可访问

4. 答：父类中的所有成员均可以被继承下来，只是私有成员继承后不能被访问。
5. 答：定义子类的构造函数时，一般需要指定其父类的构造形式，调用何种父类的构造函数。对父类部分的初始化可以在子类的构造函数初始化列表中进行传递参数。
6. 答：虚基类是用于解决同一个父类在子类中多的多份拷贝，而引起歧义。
7. 答：赋值兼容规则是指在需要基类对象的任何地方都可以使用公有派生类的对象来替代。赋值兼容规则中所指的替代包括以下的情况：
- (1) 派生类的对象可以赋值给基类对象。
 - (2) 派生类的对象可以初始化基类的引用。
 - (3) 派生类对象的地址可以赋给指向基类的指针。

8. 答：参考代码如下：

```
#include <iostream.h>
class A
{
    int a,b;
public:
    A(int a1,int b2)
    {
        a=a1;
        b=b2;
        cout<<"构造 A"<<endl;
    }
    ~A()
    {
        cout<<"析构 A"<<endl;
    }
    int geta()
    {
        return a;
    }
    int getb()
    {
        return b;
    }
};
class B:public A
{
public:
    B(int a,int b):A(a,b)
```

```

    {
        cout<<"构造 B"<<endl;
    }
    ~B()
    {
        cout<<"析构 B"<<endl;
    }
    int geta()
    {
        return 2*A::geta();
    }
};
void main()
{

```

```

    B o(10,20);
    A *p=&o;
    cout<<p->geta();
}

```

9. 答：参考代码如下：

```

#include <iostream.h>
class A
{
public:
    A()
    {
        cout<<"构造 A"<<endl;
    }
};
class B
{
public:
    B()
    {
        cout<<"构造 B"<<endl;
    }
};
class C:public A
{
    B mb;
};
void main()
{
    C o;
}

```

10. 答:

(1)

```
cout<<c.GetB()<<endl;//错误
cout<<d.GetB()<<endl;//错误
注释错误行后的运行结果如下:
```

```
0
0
8
12
```

(2)

```
void f1()
{
    a1++; //错误
}
b++; //错误
a1++; //错误
f1(); //错误
f2();
f3();
```

11. 答: 程序运行结果:

构造 A:1

构造 B:2

构造 C:

析构 C:

析构 B:2

析构 A:1

第 10 章习题答案

1. 答: 编译时多态是在编译的过程中确定了同名操作的具体操作对象, 运行时多态是在程序运行过程中才动态的确定操作所指的具体对象。

2. 答: 虚函数在函数的返回类型前使用 `virtual` 声明, 使用虚函数, 必须注意下列事项:

(1) 只有类的成员函数才能说明为虚函数。这是因为虚函数仅适用于有继承关系的类对象, 所以普通函数不能说明为虚函数。

(2) 静态成员函数不能是虚函数, 因为静态成员函数不受限于某个对象, 而虚函数的调用必须依附于某一个对象。

(3) 内联函数不能是虚函数, 因为内联函数是不能在运行中动态确定其位置的。即使虚函数在类的内部定义, 编译时, 仍将其看作非内联的。

(4) 构造函数不能是虚函数, 因为构造时, 对象还是一片未定型的空间。只有在构造完成后, 对象才能成为一个类的名副其实的实例。

(5) 析构函数可以是虚函数, 而且通常声明为虚函数。例如, 当基类对象和子类对象以不同

方式申请了堆空间。

3. 答：纯虚函数是没有函数体的虚函数，以“=0”生命？抽象类是指含有纯虚函数的类。

4. 答：运算符“++”的前置重载形式，后置重载声明形式如下：

```
class X
{
    //.....
public:
    X operator++(); //前置
    X operator++(int); //后置
    //.....
};
```

5. 答：参考第9章代码。

6. 答：参考代码如下：

```
class Object
{
public:
    virtual double CalArea()=0;
};
```

```
class Rect:public Object
{
```

```
    int x,y;
public:
    Rect(int a,int b)
    {
        x=a;
        y=b;
    }
    virtual double CalArea()
    {
        return x*y;
    }
};
```

```
class Circle:public Object
```

```
{
    int r;
public:
    Circle(int a)
    {
        r=a;
    }
    virtual double CalArea()
    {
        return 3.14*r*r;
    }
};
```

```
};
```

7. 答：参考代码如下：

```
#include <iostream.h>
```

```
class shape
```

```
{
```

```
public:
```

```
    virtual void draw()
```

```
    {
```

```
    }
```

```
};
```

```
class circle:public shape
```

```
{
```

```
public:
```

```
    virtual void draw()
```

```
    {
```

```
        cout<<"绘制 circle"<<endl;
```

```
    }
```

```
};
```

```
class squre:public shape
```

```
{
```

```
public:
```

```
    virtual void draw()
```

```
    {
```

```
        cout<<"绘制 squre"<<endl;
```

```
    }
```

```
};
```

```
class triangle:public shape
```

```
{
```

```
public:
```

```
    virtual void draw()
```

```
    {
```

```
        cout<<"绘制 triangle"<<endl;
```

```
    }
```

```
};
```

```
void main()
```

```
{
```

```
    shape *p;
```

```
    circle c;
```

```
    p=&c;
```

```
    p->draw();
```

```
    squre s;
```

```
    p=&s;
```

```
    p->draw();
```

```
    triangle t;
```

```
p=&t;  
p->draw();  
}
```

8. 答：程序运行结果：

(1)

构造 A

构造 A

构造 B

class A!

class B!

class A!

class B!

析构 B

析构 A

析构 A

(2)

A 的值为：0,0

A 的值为：1,1

A 的值为：2,2

A 的值为：1,1

A 的值为：0,0

(3)

A vehicle is running!

A vehicle has stopped!

A bicycle is running!

A bicycle has stopped!

A motorcar is running!

A motorcar has stopped!

A motorcycle is running!

A motorcycle has stopped!

A vehicle is running!

A bicycle is running!

A motorcar is running!

A motorcycle is running!

9. 答：

b.f2(); //错误

pa->f1(2); //错误

pa->f2(2); //错误

注释上述三行后，程序运行结果如下：

VAF

AF1

VAF2

VBF

BF1 1

BF1 2

VBF2

VBF

AF1

VAF2

第 11 章习题答案

1.参考代码:

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
    double a,b,c,s,area;
    cout<<"please input a,b,c:";
    cin>>a>>b>>c;
    if (a+b<=c)
        cerr<<"a+b<=c,error!"<<endl;
    else if(b+c<=a)
        cerr<<"b+c<=a,error!"<<endl;
    else if (c+a<=b)
        cerr<<"c+a<=b,error!"<<endl;
    else
    {
        s=(a+b+c)/2;
        area=sqrt(s*(s-a)*(s-b)*(s-c));
        cout<<"area="<<area<<endl;
    }
    return 0;
}
```

2.参考代码:

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    float a[5];
    cout<<"input data:";
    for(int i=0;i<5;i++)
        cin>>a[i];
    cout<<setiosflags(ios::fixed)<<setprecision(2);
```

```

    for(i=0;i<5;i++)
        cout<<setw(10)<<a[i]<<endl;
    return 0;
}

```

3. 参考代码:

```

#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    for(int n=1;n<8;n++)
        cout<<setw(20-n)<<setfill(' ')<<" "
        <<setw(2*n-1)<<setfill('B')<<"B"<<endl;
    return 0;
}

```

4. 参考代码:

```

#include <iostream>
#include <fstream>
using namespace std;
void main()
{
    int a[10];
    ofstream outfile1("f1.dat"),outfile2("f2.dat");
    if(!outfile1)
    {
        cerr<<"open f1.dat error!"<<endl;
        exit(1);
    }
    if(!outfile2)
    {
        cerr<<"open f2.dat error!"<<endl;
        exit(1);
    }
    cout<<"enter 10 integer numbers:"<<endl;
    for(int i=0;i<10;i++)
    {
        cin>>a[i];
        outfile1<<a[i]<<" ";
    }
    cout<<"enter 10 integer numbers:"<<endl;
    for(i=0;i<10;i++)
    {
        cin>>a[i];
        outfile2<<a[i]<<" ";
    }
}

```

```

        outfile1.close();
        outfile2.close();
    }
5 参考代码:
#include <iostream>
#include <fstream>
using namespace std;
struct staff
{
    int num;
    char name[20];
    int age;
    double pay;
};

int main()
{
    staff staf[7]={2101,"Li",34,1203,2104,"Wang",23,674.5,2108,"Fun",54,778,
        3006,"Xue",45,476.5,5101,"Ling",39,656.6},staf1;
    fstream iofile("staff.dat",ios::in|ios::out|ios::binary);
    if(!iofile)
    {
        cerr<<"open error!"<<endl;
        abort();
    }
    int i,m,num;
    cout<<"Five staff :"<<endl;
    for(i=0;i<5;i++)
    {
        cout<<staf[i].num<<" "<<staf[i].name<<" "<<staf[i].age<<" "<<staf[i].pay<<endl;
        iofile.write((char *)&staf[i],sizeof(staf[i]));
    }
    cout<<"please input data you want insert:"<<endl;
    for(i=0;i<2;i++)
    {
        cin>>staf1.num>>staf1.name>>staf1.age>>staf1.pay;
        iofile.seekp(0,ios::end);
        iofile.write((char *)&staf1,sizeof(staf1));
    }
    iofile.seekg(0,ios::beg);
    for(i=0;i<7;i++)
    {
        iofile.read((char *)&staf[i],sizeof(staf[i]));
        cout<<staf[i].num<<" "<<staf[i].name<<" "<<staf[i].age<<" "<<staf[i].pay<<endl;
    }
}

```

```

bool find;
cout<<"enter number you want search,enter 0 to stop.";
cin>>num;
while(num)
{
    find=false;
    ifstream.seekg(0,ios::beg);
    for(i=0;i<7;i++)
    {
        ifstream.read((char *)&staf[i],sizeof(staf[i]));
        if(num==staf[i].num)
        {
            m=ifstream.tellg();
            cout<<num<<" is No."<<m/sizeof(staf1)<<endl;
            cout<<staf[i].num<<" "<<staf[i].name<<" "
                <<staf[i].age<<" "<<staf[i].pay<<endl;
            find=true;
            break;
        }
    }
    if(!find)
        cout<<"can't find "<<num<<endl;
    cout<<"enter number you want search,enter 0 to stop.";
    cin>>num;
}
ifstream.close();
return 0;
}

```

6. 答：数据从一个对象到另一个对象的传送被抽象为“流”。从流中获取数据的操作称为提取操作，向流中添加数据的操作称为插入操作。操作系统是将键盘、屏幕、打印机和通信端口作为扩充文件来处理的，而这种处理是通过操作系统的设备驱动程序来实现的，因此，从C++程序员的角度来看，这些设备与磁盘文件是等同的，I/O 流类就是用来与这些扩充文件进行交互。

7. 答：clog 流对象也是标准错误流，它的作用和 cerr 相同，都是在显示器显示出错信息。区别在于，cerr 是不经过缓冲区，直接向显示器上输出有关信息，而 clog 中的信息存放在缓冲区中，缓冲区满后或遇到 endl 时向显示器输出。

8. 参考代码：

```

#include "iostream.h"
#include "fstream.h"
void main()
{
    ofstream myfile;
    myfile.open("c:\\test1.txt",ios::in);
    myfile<<"已成功写入文件!";
}

```

```
}
```

9. 参考代码:

```
#include "iostream.h"
#include "fstream.h"
void main()
{
    ifstream myfile;
    char line[100];
    myfile.open("c:\\test1.txt",ios::in);
    myfile.getline(line,100,'\t');
    cout<<line;
}
```

10. 参考代码:

```
#include "iostream.h"
#include "fstream.h"
void main()
{
    ofstream myofile;
    myofile.open("c:\\test1.txt",ios::app);
    myofile<<"已成功添加字符!";
    myofile.close();
    ifstream myifile;
    char line[100];
    myifile.open("c:\\test1.txt",ios::in);
    myifile.getline(line,100,'\t');
    cout<<line;
    myifile.close();
}
```

11. 参考代码:

```
#include "iostream.h"
#include "fstream.h"
class dog
{
    private:
        float w;
        int age;
    public:
        dog()
        {
            w=0.0;
            age=0;
        }
        float getw();
        int getage();
}
```

```

        void setw(float w1);
        void setage(int age1);
};
float dog::getw()
{
    return w;
}
int dog::getage()
{
    return age;
}
void dog::setw(float w1)
{
    w=w1;
}
void dog::setage(int age1)
{
    age=age1;
}
void main()
{
    dog dog1;
    char c[10];
    dog1.setw(25.3f);
    dog1.setage(25);
    ofstream myfile;
    myfile.open("c:\\test1.txt",ios::in);
    myfile<<dog1.getage();
    myfile<<dog1.getw();
    myfile.close();
}

```

第 12 章习题答案

1. 答：模板函数，模板类
2. 答：template <>
3. 答：参数类型
4. 答：STL
5. 答：当编译系统发现了一个对应的函数调用时，将根据实参的类型来确认是否匹配函数模板中对应的形参，然后生成一个重载函数，称该重载函数为模板函数。函数模板与模板函数的区别：二者区别可以类比类与对象的区别。

6. 答：函数模板与类相似是模板的定义，而模板函数与对象相似。是函数模板的实例，具有程序代码，占用内存空间。说明了一个类模板后，也可以创建类模板的实例即生成模板类。类模板与模板类的区别是：类模板是模板的定义，不是一个实在的类，模板类才是实实在在的类。

7. 答：

程序一：

队列 A 的元素为：1 3 5 7 9

程序二：

数组的值为：{ 5, 2, 7, 4, 4, 2, 6, 1}

输入想查找的数：5

最先等于 5 的是第 1 个元素

排序前的结果：

5 2 7 4 4 2 6 1

升序排列后的结果：

1 2 2 4 4 5 6 7

降序排列后的结果：

7 6 5 4 4 2 2 1

第 13 章习题答案

1. 参考代码：

```
#include <iostream>
#include <cmath>
using namespace std;
double q(double,double,double);
void main()
{
    double a,b,c,p,x1,x2;
    cout<<"please enter a,b,c:";
    cin>>a>>b>>c;
    p=-b/(2*a);
    try
    {
        x1=p+q(a,b,c);
        x2=p-q(a,b,c);
        cout<<"x1="<<x1<<endl<<"x2="<<x2<<endl;
    }
    catch(double d)
    {
        cout<<"a="<<a<<","b="<<b<<","c="<<c<<","disc="<<d<<","error!"<<endl;
    }
    cout<<"end"<<endl;
```

```

}
double q(double a,double b,double c)
{
    double disc;
    disc=b*b-4*a*c;
    if (disc<0) throw disc;
    return sqrt(disc)/(2*a);
}

```

2.答:

```

main begift
call fun()
constructor-1101
1101,Taft
constructor-1102
1102,Li
destructor-1102
destructor-1101
main end

```

3.参考代码:

```

#include <iostream>
#include <string>
using namespace std;
class Student
{
public:
    Student(int n,string nam)
    {cout<<"constructor-"<<n<<endl;
    num=n;name=nam;}
    ~Student()
    {
        cout<<"destructor-"<<num<<endl;}
    void get_data();
private:
    int num;
    string name;
};
void Student::get_data()
{
    if(num==0) throw num;
    else cout<<num<<" "<<name<<endl;
    cout<<"in get_data()"<<endl;
}
void fun()
{

```



```

Student stud1(1101,"tan");
stud1.get_data();
try
{
    Student stud2(0,"Li");
    stud2.get_data();
}
catch(int n)
{
    cout<<"num="<<n<<"error!"<<endl;
}
}
int main()
{
    cout<<"main begin"<<endl;
    cout<<"call fun()"<<endl;
    fun();
    cout<<"main end"<<endl;
    return 0;
}

```

4. 答：在编程过程中，很多时候我们是无法确定一段代码是否总是能够正常工作的，或者因为程序访问了并不存在的资源，或者由于一些变量超出了预期的范围等等，这些情况我们统称为出错（异常）。异常处理是对这些异常情况进行处理的程序，能够使程序合理化的结束。

5. 答：C++的异常处理机制使得异常的引发和处理不必在同一函数中，这样底层的函数可以着重解决具体问题，而不必过多地考虑对异常的处理。上层调用者可以在适当的位置设计针对不同类型异常的处理。

6. 答：如果某段程序中发现了自己不能处理的异常，就可以使用 **throw** 表达式抛出这个异常，将它抛给调用者。**try** 子句后的复合语句是代码的保护段。如果预料某段程序代码（或对某个函数的调用）有可能发生异常，就将其放到 **try** 子句之后。如果这段代码（或被调用函数）运行时真的遇到异常情况，其中的 **throw** 表达式就会抛出这个异常。**catch** 子句后的复合语句是异常处理程序，捕获由 **throw** 表达式抛出的异常。异常类型声明部分指明了子句处理的异常类型，它与函数的型参是类似的，可以是某个类型的值，也可以是引用。类型可以使任何有效的数据类型，包括 C++ 类。当异常被抛出以后，**catch** 子句便依次被检查，若某个 **catch** 子句的异常类型声明与被抛出的异常类型一致，则执行该段异常处理程序。如果异常类型声明是一个省略号（...），**catch** 子句便处理任何类型的那个异常，这段处理程序必须是 **try** 块的最后一段处理程序。

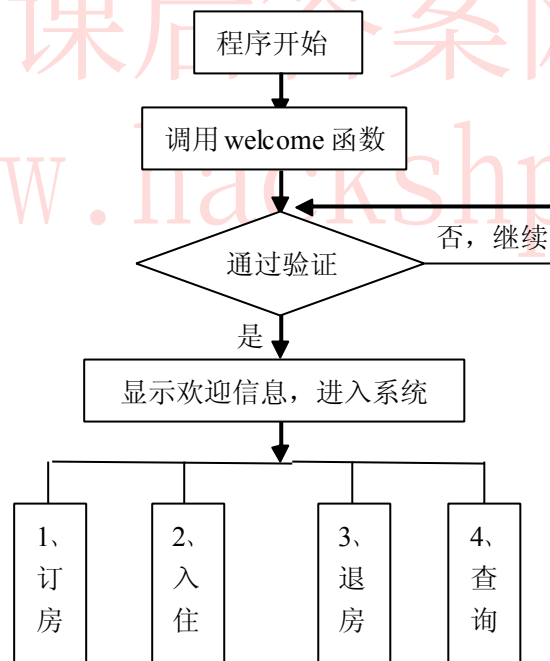
7. 答：不是，因为程序发生异常，就表示程序就中断程序的执行，转到异常处理程序继续执行。

8. 答：一些 C++ 标准语言库中的函数也会抛出一些例外，我们可以用 **try** 语句来捕获它们。这些异常抛出的参数都是 **std::exception** 引申出的子类类型的。这个类(**std::exception**) 被定义在 C++ 标准头文件中，用来作为 **exceptions** 标准结构的模型。这是一个类结构，如果你包括了一个 **catch** 语句块使用地址来捕获这个结构中的任意一种例外(也就是说在类型后面加地址符&)，你同时可以捕获所有引申类的异常(C++的继承原则)。

第 14 章习题答案

1. 答:

系统分析: 作为一个宾馆客房间管理系统, 必然涉及到客房对象, 顾客对象等, 系统设置了 80 个房间, 其中房间分四个等级, 每个等级的价格是不一样的, 可以通过房间编号得到房间的等级。然后就是房间的分配问题, 就是当有顾客要求定房或入住时, 程序能够根据用户要求入住的等级到相应等级的房间中去查找一个还没有预定的房间和一个没有入住的空闲房间。程序首先调用 `initial_room` 函数初始化 80 个房间的信息, 包括房间编号, 房间等级, 房间价格, 房间状态。其中房间编号和房间等级有直接联系, 只要知道了房间编号就可以通过计算得到该房间的等级, 房间状态初始化时等于 0, 表示该房间既没有被预定, 也没有被入住。然后调用 `welcome` 函数, 考虑到作为一个宾馆管理系统的安全性, 对操作本系统的人员也要求认证的, 其中在本函数中就要求只有通过输入了正确的用户名和密码才能操作系统在这里, 系统给操作员显示一个操作菜单如: 1->定房 2->入住 3->退房 4->查询, 当操作员选择不同的数字就实现不同的功能。



参考代码如下:

```
#include<iostream.h>
#include<string.h>
#include<fstream>
#include<vector>
void initial_room();//初始化 80 个房间信息
void welcome();      //验证用户登陆,登陆成功显示欢迎信息
void enter();        //入住
void book_room();    //预定房间
```

```

void check_in();    //登记
void check_out();  //登出
void inquire();    //查询
int i=0;
struct Room
{
    int number;    //
    int dank;
    int price;    //价钱
    int state;    //状态
};
class Customer
{
public:
    Customer();
    void set_name(char *n){strcpy(name,n);}    //名字
    void set_ID(char *p){strcpy(ID,p);}    //身份证号
    void set_room_number(int n){room_number=n;}    //房间号
    void set_day(int d){day=d;}    //入住天数
    void set_prepaid(int p){prepaid=p;}    //
    char *get_name(){return name;}    //接收名字
    char *get_ID(){return ID;}    //接收身份证号
    int get_room_number(){return room_number;}    //接收房号
    int get_day(){return day;}    //接收日子
    int get_prepaid(){return prepaid;}    //接收预付款
    virtual ~Customer();
private:
    char name[10],ID[19];    //名字
    int room_number;    //房间号
    int prepaid;    //预付款
    int change;    //变换房间
    int day;    //入住天数
};
Customer::Customer()
{
    cout<<"请您输入顾客的姓名\n";
    cin>>name;
    cout<<"请您输入顾客的身份证号码\n";
    cin>>ID;
    prepaid=change=day=0;    //初始化基本状态为空
}
Customer::~~Customer()    //清空函数
{
    cout<<"该顾客退房成功!"<<endl;
}

```

```

}
Room room[80];
Customer *customer[80];           //指针数组

void main()                       //程序入口从这里开始运行
{
    char choice='1';              //判断选择项
    initial_room();//初始化 80 个房间的信息,分四个等级
    welcome();                    //验证用户登陆,登陆成功显示欢迎信息
    while(choice=='1')
    {
        enter();                //根据用户的选择执行不同的功能
        cout<<endl;
        cout<<"继续使用本系统请按\"1\",退出请按\"2\"!  ";
        cin>>choice;
        cout<<endl;
    }
}
//系统登陆界面
void welcome()                   //验证用户登陆,登陆成功显示欢迎信息
{
    char name[10],code[15];
    cout<<"请输入用户名和密码(以空格隔开):\n";
    cin>>name>>code;
    while((strcmp(name,"abc")!=0)||((strcmp(code,"888")!=0))
    {
        cout<<"用户名或密码输入有误,请重新输入!\n";
        cin>>name>>code;
    }
}

```

```

cout<<endl;
cout<<"                88      88      8888      8888888888      88888888      8
"<<endl;
    cout<<"                88      88      88      88      8      8      8
"<<endl;
    cout<<"                88      88      88      88      8      8      8
"<<endl;
    cout<<"                8888888888      88      88      8      88888888      8
"<<endl;
    cout<<"                88      88      88      88      8      8      8
"<<endl;
    cout<<"                88      88      88      88      8      8      8
"<<endl;
    cout<<"                88      88      88      88      8      8      8

```

```

"<<endl;
    cout<<"          88      88      8888      8      88888888
88888888      "<<endl;
    cout<<"
*****"<<endl;
    cout<<endl;
    cout<<"          欢迎登入客房管理系统(1.0)
"<<endl;
    cout<<endl;
    cout<<"
*****"<<endl;
}
//系统入口
void enter()
{
    int kind_of_service;
    cout<<"          请选择服务类别:按回车键确认: \n";
    cout<<"          (1)、定房 (2)、入住 (3)、退房 (4)、查询: ";
    cin>>kind_of_service; //服务种类
    if((kind_of_service>4) || (kind_of_service<1)) //服务条件判断
    {
        cout<<"您的输入有误, 请重试!\n";
        cin>>kind_of_service;
    }
    else
    {
        switch(kind_of_service)
        {
            case 1: book_room();break;
            case 2: check_in();break;
            case 3: check_out();break;
            case 4: inquire();break;
        }
    }
}
//订房实现
void book_room()
{
    customer[i]=new Customer;
    int room_standard,day;
    cout<<"请您选择预定房间的标准:\n";
    cout<<" 1.单人间/天 150 元\n";
    cout<<" 2.双人间/天 200 元\n";
    cout<<" 3.标准间/天 300 元\n";
    cout<<" 4.总统套房/天 600 元\n";
    cin>>room_standard;

```

```

cout<<"请输入预定天数\n";
cin>>day;
customer[i]->set_day(day);
switch(room_standard)
{
    int n;
case 1:
    cout<<"住房费用总共为: "<<day*150<<"元\n";    //计算住房费用
    for(n=0;n<20;n++)    //从第一个级别的房间中查找一个空闲的房间
    {
        if(room[n].state==0)    //state=0 表示该住房没有被预定的
        {
            cout<<"预定成功★房间号码为: "<<room[n].number<<endl;
            room[n].state=1;
            customer[i]->set_room_number(room[n].number);
            break;
        }
    }
    break;
case 2:
    cout<<"住房费用总共为: "<<day*200<<" 元\n";
    for(n=20;n<40;n++)
    {
        if(room[n].state==0)
        {
            cout<<"预定成功★房间号码为: "<<room[n].number<<endl;
            room[n].state=1;
            customer[i]->set_room_number(room[n].number);
            break;
        }
    }
    break;
case 3:
    cout<<"住房费用总共为: "<<day*300<<" 元\n";
    for(n=40;n<60;n++)
    {
        if(room[n].state==0)
        {
            cout<<"预定成功★房间号码为: "<<room[n].number<<endl;
            room[n].state=1;
            customer[i]->set_room_number(room[n].number);
            break;
        }
    }
}

```

```

        break;
    case 4:
        cout<<"住房费用总共为: "<<day*600<<"元\n";
        for(n=60;n<80;n++)
        {
            if(room[n].state==0)
            {
                cout<<"预定成功★房间号码为: "<<room[n].number<<endl;
                room[n].state=1;
                customer[i]->set_room_number(room[n].number);
                break;
            }
        }
        break;
    }
    i++; //住房的顾客数加 1
}
//入住登记
void check_in()
{
    char name1[10],id1[19];
    int ding_or_no,prepaid;
    cout<<"该顾客订房了吗? (1->订了 2->没订) ";
    cin>>ding_or_no;
    if(ding_or_no==1)
    {
        cout<<"请输入顾客的姓名:"<<endl; //用一维字符数组接收姓名
        cin>>name1;
        cout<<"请输入顾客的身份证号码:"<<endl; //用一维字符数组接收身份证号码
        cin>>id1;
        for(int j=0;j<=i;j++)
        {
            if((strcmp(customer[j]->get_name(),name1)==0)&&(strcmp(customer[j]->get_ID(),id1)==0))
            //查找该顾客的定房信息
            {
                int num=customer[j]->get_room_number(); //取得定房的房间号
                cout<<"顾客"<<name1<<"今天入住本宾馆! 房间号码为: "<<num<<endl;
                switch(num/100)
                {
                    case 6:
                        prepaid=customer[j]->get_day()*150;
                        customer[j]->set_prepaid(prepaid);
                        cout<<"请收房费 "<<prepaid<<"元整!"<<endl; //开始收费

```

```

        room[num%100-1].state=2;    //修改房间状态为入住状态
        break;
    case 7:
        prepaid=customer[j]->get_day()*200;
        customer[j]->set_prepaid(prepaid);
        cout<<"请收房费 "<<prepaid<<"元整!"<<endl;
        room[19+num%100].state=2;
        break;
    case 8:
        prepaid=customer[j]->get_day()*300;
        customer[j]->set_prepaid(prepaid);
        cout<<"请收房费 "<<prepaid<<"元整!"<<endl;
        room[39+num%100].state=2;
        break;
    case 9:
        prepaid=customer[j]->get_day()*600;
        customer[j]->set_prepaid(prepaid);
        cout<<"请收房费 "<<prepaid<<"元整!"<<endl;
        room[59+num%100].state=2;
        break;
    }
    break;
}
}
else
{
    customer[i]=new Customer;
    int room_standard,day;
    cout<<"请您选择预定房间的标准:\n";
    cout<<"1.单人间/天 150 元\n2.双人间/天 200 元\n3.标准间/天 300 元\n4.总统套房/天 600
元\n";
    cin>>room_standard;
    cout<<"请输入住宿天数\n";
    cin>>day;
    customer[i]->set_day(day);
    switch(room_standard)
    {
        int n;
    case 1:
        prepaid=day*150;
        customer[i]->set_prepaid(prepaid);
        cout<<"请收房费 "<<prepaid<<"元整!\n";

```



```

for(n=0;n<20;n++)
{
    if(room[n].state==0)
    {
        cout<<"入住房间号码为: "<<room[n].number<<endl;
        room[n].state=2;
        customer[i]->set_room_number(room[n].number);
        break;
    }
}
break;
case 2:
    prepaid=day*200;
    customer[i]->set_prepaid(prepaid);
    cout<<"请收房费 " <<prepaid<<"元整!\n";
    for(n=20;n<40;n++)
    {
        if(room[n].state==0)
        {
            cout<<"入住房间号码为: "<<room[n].number<<endl;
            room[n].state=2;
            customer[i]->set_room_number(room[n].number);
            break;
        }
    }
    break;
case 3:
    prepaid=day*300;
    customer[i]->set_prepaid(prepaid);
    cout<<"请收房费 " <<prepaid<<"元整!\n";
    for(n=40;n<60;n++)
    {
        if(room[n].state==0)
        {
            cout<<"入住房间号码为: "<<room[n].number<<endl;
            room[n].state=2;
            customer[i]->set_room_number(room[n].number);
            break;
        }
    }
    break;
case 4:
    prepaid=day*600;
    customer[i]->set_prepaid(prepaid);

```

```

        cout<<"请收房费 "<<prepaid<<"元整!\n";
        for(n=60;n<80;n++)
        {
            if(room[n].state==0)
            {
                cout<<"入住房间号码为: "<<room[n].number<<endl;
                room[n].state=2;
                customer[i]->set_room_number(room[n].number);
                break;
            }
        }
        break;
    }
    i++;
}
}
//退房结算
void check_out()
{
    char name2[10],id2[19];
    int standard,j,room_number,day1,day2,day;
    cout<<"请输入要退房的顾客姓名和身份证号码:\n";
    cin>>name2>>id2;
    cout<<"请输入该顾客实住天数:\n";
    cin>>day2;
    for(j=0;j<i;j++)
    {
        if((strcmp(customer[j]->get_name(),name2)==0)&&(strcmp(customer[j]->get_ID(),id2)==0)
        )
        {
            room_number=customer[j]->get_room_number();
            standard=room_number/100;
            day1=customer[j]->get_day();
            day=day1-day2;
            switch(standard)
            {
                case 6:
                    cout<<"顾客的房间号是"<<room_number<<":为单人间,每天 150 元\n";
                    cout<<"该顾客预付了房费 "<<customer[j]->get_prepaid()<<"元, 实际消费
                    "<<day2*150<<"元整!\n";
                    cout<<endl;
                    if(day>0)
                        cout<<"请退给该顾客 "<<day*150<<" 元整!\n";
                    if(day<0)

```

```

        cout<<"请补收该顾客住房费 "<<-day*150<<" 元整!\n";
        break;
    case 7:
        cout<<"顾客的房间号是"<<room_number<<":为单人间,每天 200 元\n";
        cout<<"该顾客预付了房费 "<<customer[j]->get_prepaid()<<" 元, 实际消费
"<<day*200<<"元整!\n";
        cout<<endl;
        if(day>0)
            cout<<"请退给该顾客 "<<day*200<<" 元整!\n";
        if(day<0)
            cout<<"请补收该顾客住房费 "<<-day*200<<" 元整!\n";
        break;
    case 8:
        cout<<"顾客的房间号是"<<room_number<<":为单人间,每天 300 元\n";
        cout<<"该顾客预付了房费 "<<customer[j]->get_prepaid()
<<"元, 实际消费 "<<day*300<<"元整!\n";
        cout<<endl;
        if(day>0)
            cout<<"请退给该顾客 "<<day*300<<" 元整!\n";
        if(day<0)
            cout<<"请补收该顾客住房费 "<<-day*300<<" 元整!\n";
        break;
    case 9:
        cout<<"顾客的房间号是"<<room_number<<":为单人间,每天 600 元\n";
        cout<<"该顾客预付了房费 "<<customer[j]->get_prepaid()
<<"元, 实际消费 "<<day*600<<"元整!\n";
        cout<<endl;
        if(day>0)
            cout<<"请退给该顾客 "<<day*600<<" 元整!\n";
        if(day<0)
            cout<<"请补收该顾客住房费 "<<-day*600<<" 元整!\n";
        break;
    }
    cout<<endl;
    cout<<"退房核算清楚,请按 1: ";
    char account;
    cin>>account;
    if(account=='1')
    {
        for(int k=0;k<80;k++)
        {
            if(room[k].number==customer[j]->get_room_number())
                room[k].state=0;
        }
    }

```

```

        i--;
        for(;j<i;j++)
        {
            customer[j]=customer[j+1];
        }
        delete customer[i];
    }
}
}
//信息查询
void inquire()
{
    char inquire_choice;
    cout<<"房间信息查询请按 1, 顾客信息查询请按 2: "<<endl;
    cin>>inquire_choice;
    if(inquire_choice=='1')
    {
        int j,k=0;
        cout<<endl;
        cout<<"下列房间还没人预订, 也没人入住: "<<endl;
        for(j=0;j<80;j++)
        {
            if(room[j].state==0)
            {
                if(k%10==0) cout<<endl;
                cout<<room[j].number<<"\t";
                k++;
            }
        }
        cout<<endl;
        cout<<endl;
        k=0;
        cout<<"下列房间已预订: "<<endl;
        for(j=0;j<80;j++)
        {
            if(room[j].state==1)
            {
                if(k%10==0) cout<<endl;
                cout<<room[j].number<<"\t";
                k++;
            }
        }
        k=0;
    }
}

```

```

cout<<endl;
cout<<endl;
cout<<"下列房间有人入住: "<<endl;
for(j=0;j<80;j++)
{
    if(room[j].state==2)
    {
        if(k%10==0) cout<<endl;
        cout<<room[j].number<<"\t";
        k++;
    }
}
cout<<endl;
cout<<endl;
}
else if(inquire_choice=='2')
{
    cout<<"按姓名查询请按 1, 按身份证查询请按 2:"<<endl;
    char inquire_choice;
    cin>>inquire_choice;
    if(inquire_choice=='1')
    {
        char name3[10];
        cout<<"请输入顾客的姓名: "<<endl;
        cin>>name3;
        for(int j=0;j<=i;j++)
        {
            if(strcmp(customer[j]->get_name(),name3)==0)
            {
                cout<<name3<<"的住房信息如下:\n";
                cout<<"\t"<<"房间号为: "<<customer[j]->get_room_number()<<endl;
                cout<<"\t"<<"预付房费为: "<<customer[j]->get_prepaid()<<endl;
            }
        }
    }
}
if(inquire_choice=='2')
{
    char id3[10];
    cout<<"请输入顾客的姓名: "<<endl;
    cin>>id3;
    for(int j=0;j<=i;j++)
    {
        if(strcmp(customer[j]->get_ID(),id3)==0)
        {

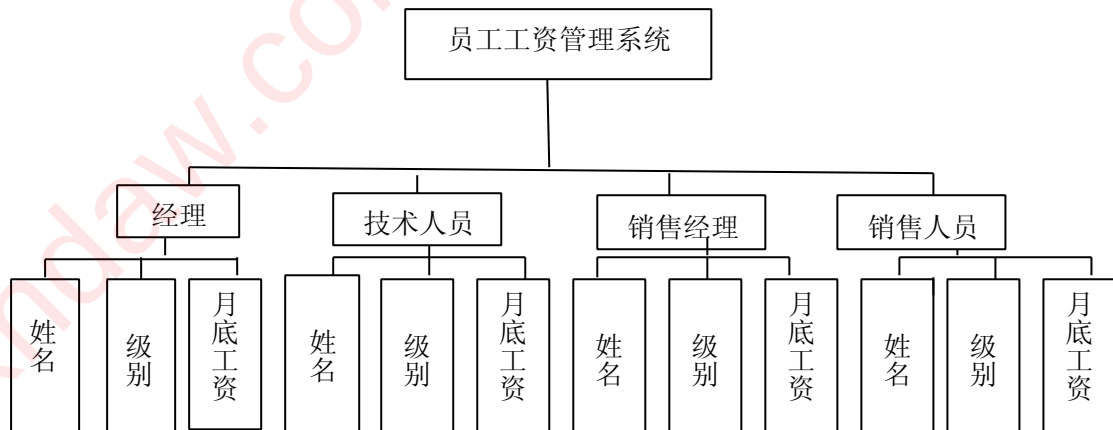
```

```

        cout<<customer[j]->get_name()<<"的住房信息如下:\n";
        cout<<"\t"<<"房间号为: "<<customer[j]->get_room_number()<<endl;
        cout<<"\t"<<"预付房费为: "<<customer[j]->get_prepaid()<<endl;
    }
}
}
}
//房间信息初始化
void initial_room() //对 80 个房间进行初始化,分 4 个等级
{
    int j,k=601;
    for(j=0;j<20;j++)
    {
        room[j].number=k++;
        room[j].dank=1;
        room[j].price=150;
        room[j].state=0;
    }
    k=701;
    for(j=20;j<40;j++)
    {
        room[j].number=k++;
        room[j].dank=2;
        room[j].price=200;
        room[j].state=0;
    }
    k=801;
    for(j=40;j<60;j++)
    {
        room[j].number=k++;
        room[j].dank=3;
        room[j].price=300;
        room[j].state=0;
    }
    k=901;
    for(j=60;j<80;j++)
    {
        room[j].number=k++;
        room[j].dank=4;
        room[j].price=600;
        room[j].state=0;
    }
}
}

```

2. 答：对系统进行初步分析后，得到系统的功能模块图如下图所示。



系统参考代码如下：

```
#include<iostream.h>
```

```
#include<string.h>
```

```
class employee
```

```
{
```

```
protected:
```

```
    char *name; //姓名
```

```
    int individualempNo; //编号
```

```
    int grade; //级别
```

```
    float accumpay; //月薪总额
```

```
    static int employeeNo; //本公司职员编号目前最大值
```

```
public:
```

```
    employee(); //构造函数
```

```
    ~employee(); //析构函数
```

```
    virtual void pay()=0; //计算月薪函数(纯虚函数)
```

```
    virtual void promote(int increment=0); // 升级函数（虚函数）
```

```
    virtual void displaystatus()=0; //纯虚函数
```

```
};
```

```
class technician:public employee //技术人员类
```

```
{
```

```
private:
```

```
    float hourlyrate; //每小时酬金
```

```
    int workhours; //当月工时数
```

```
public:
```

```
    technician(); //构造函数
```

```
    void promote(int); //升级函数
```

```
    void pay(); //计算月薪函数
```

```
    void displaystatus();
```

```
};
```

```

class manager:virtual public employee //经理类
{
protected:
    float monthllypay; //固定月薪数
public:
    manager(); //构造函数
    void promote(int); //升级函数
    void pay(); //计算月薪函数
    void displaystatus();
};
class salesman:virtual public employee //销售人员类
{
protected:
    float commrate; //按销售额提取酬金的百分比
    float sales; //当月销售额
public:
    salesman(); //构造函数
    void promote(int); //升级函数
    void pay(); //计算月薪函数
    void displaystatus();
};
class salesmanager:public manager,public salesman //销售经理类
{
public:
    salesmanager(); //构造函数
    void promote(int); //升级函数
    void pay(); //计算月薪函数
    void displaystatus();
};
int employee::employeeNo=1000; //员工编号基数为 1000
employee::employee()
{
    char namestr[50]; //输入员工姓名时首先临时存放在 namestr 中
    cout<<"请输入一个雇员的姓名:";
    cin>>namestr;
    name=new char[strlen(namestr)+1]; //动态创建对象
    strcpy(name,namestr); //字符串的拷贝
    individualempNo=employeeNo++; //新输入的员工编号为目前最大编号加一
    grade=1; //级别初值为 1
    accumpay=0.0; //月薪总额初值为 0
}
employee::~~employee()
{
    delete[]name; //删除对象
}

```



```

}
void employee::promote(int increment)
{
    grade+=increment; //升级，提升的级别由 increment 决定
}
technician::technician()
{
    hourlyrate=100;
}
void technician::promote(int)
{
    employee::promote(2);
    //调用基类升级函数，升 2 级
}
void technician::pay()
{
    cout<<"请输入"<<name<<"本月的工作时数:";
    cin>>workhours;
    accumpay=hourlyrate*workhours;
    cout<<"兼职技术人员"<<name<<"编号"
        <<individualempNo<<"本月工资"<<accumpay<<endl;
}
void technician::displaystatus()
{
    cout<<"兼职技术人员"<<name<<"编号"<<individualempNo
        <<"级别为"<<grade<<"级，本月工资"<<accumpay<<endl;
}
salesman::salesman()
{
    commrate=0.04;
}
void salesman::promote(int)
{
    employee::promote(0);
}
void salesman::pay()
{
    cout<<"请输入"<<name<<"本月的销售额:";
    cin>>sales;
    accumpay=sales*commrate;
    cout<<"推销员"<<name<<"编号"<<individualempNo<<"本月工资"<<accumpay<<endl;
}
void salesman::displaystatus()
{
    cout<<"推销员"<<name<<"编号"<<individualempNo<<"级别为"

```

```

        <<grade<<"级， 本月工资"<<accumpay<<endl;

    }
    manager::manager()
    {
        monthllypay=8000;
    }
    void manager::promote(int)
    {
        employee::promote(3);
    }
    void manager::pay()
    {
        accumpay=monthllypay;
        cout<<"经理"<<name<<"编号"<<individualempNo<<"本月工资"<<accumpay<<endl;
    }
    void manager::displaystatus()
    {
        cout<<"经理"<<name<<"编号"<<individualempNo<<"级别为"
            <<grade<<"级， 本月工资"<<accumpay<<endl;
    }
    salesmanager::salesmanager()
    {
        monthllypay=5000;
        commrate=0.005;
    }
    void salesmanager::promote(int)
    {
        employee::promote(2);
    }
    void salesmanager::pay()
    {
        cout<<"请输入"<<employee::name<<"所管辖部门本月的销售总额";
        cin>>sales;
        accumpay=monthllypay+commrate*sales;
        cout<<"销售经理"<<name<<"编号"<<individualempNo<<"本月工资"<<accumpay<<endl;
    }
    void salesmanager::displaystatus()
    {
        cout<<"销售经理"<<name<<"编号"<<individualempNo<<"级别为"
            <<grade<<"级， 本月工资"<<accumpay<<endl;
    }
    int main()
    {

```

```

cout<<" *****" <<endl;
cout<<" 员 工 工 资 管 理 系 统" <<endl;
cout<<" *****" <<endl;
cout<<" "<<endl;
cout<<" "<<endl;
cout<<" "<<endl;
biaohao:
cout<<"*****"
*****"<<endl;
cout<<"* 0.exit 1.经理 2.技术人员 3.销售经理 4.销售人员
*"<<endl;
cout<<"*****"
*****"<<endl;
cout<<"请选择功能: "<<endl;
int a;
cin>> a;
switch(a)
{
case 0:break;
case 1:
{
manager m1;
employee* a;
a=&m1;//对对象的引用
a->promote();//指针调用函数
a->pay();
a->displaystatus();
}
goto biaohao;
break;
case 2:
{
technician t1;
employee* b;
b=&t1;
b->promote();
b->pay();
b->displaystatus();
}
goto biaohao;break;
case 3:
{
salesmanager sm1;
employee* c;

```

```

        c=&sml;
        c->promote();
        c->pay();
        c->displaystatus();
    }
    goto biaohao;break;
case 4:
    {
        salesman s1;
        employee* d;
        d=&s1;
        d->promote();
        d->pay();
        d->displaystatus();
    }
    goto biaohao;break;
default:cout<<"你输入有误，请重试！"<<endl;goto biaohao;break;
}
return 0;
}

```

课后答案网

www.hackshp.cn