1.什么是面向对象程序设计？

【解】 面向对象程序设计是一种新的程序设计范型。这种范型的主要特征是：

程序 = 对象 + 消息

面向对象程序的基本元素是对象，面向对象程序的主要结构特点是：第一，程序一般由类的定义和类的使用两部分组成；第二，程序中的一切操作都是通过向对象发送消息来实现的，对象接收到消息后，启动有关方法完成相应的操作。

面向对象程序设计方法模拟人类习惯的解题方法，代表了计算机程序设计新颖的思维方式。这种方法的提出是对软件开发方法的一场革命，是目前解决软件开发面临困难的最有希望、最有前途的方法之一。

2.什么是对象？什么是类？对象与类的关系是什么？

【解】 在现实世界中，任何事物都是对象。它可以是一个有形的具体存在的事物，例如一张桌子、一个学生、一辆汽车，甚至一个地球；它也可以是一个无形的、抽象的事件，例如一次演出、一场球赛、一次出差等。对象既可以很简单，也可以很复杂，复杂的对象可以由若干简单的对象构成，整个世界都可以认为是一个非常复杂的对象。在现实世界中，对象一般可以表示为：属性+行为，一个对象往往是由一组属性和一组行为构成的。

在面向对象程序设计中，对象是描述其属性的数据以及对这些数据施加的一组操作封装在一起构成的统一体。在C++ 中每个对象都是由数据和操作代码（通常用函数来实现）两部分组成的。

在现实世界中，“类”是一组具有相同属性和行为的对象的抽象。类和对象之间的关系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。

在面向对象程序设计中，“类”就是具有相同的数据和相同的操作（函数）的一组对象的集合，也就是说，类是对具有相同数据结构和相同操作的一类对象的描述。

类和对象之间的关系是抽象和具体的关系。类是多个对象进行综合抽象的结果，一个对象是类的一个实例。例如“学生”是一个类，它是由许多具体的学生抽象而来的一般概念。同理，桌子、教师、计算机等都是类。

3.现实世界中的对象有哪些特征？请举例说明。

【解】 现实世界中的对象，具有以下特性：

（1）每一个对象必须有一个名字以区别于其他对象；

（2）用属性来描述它的某些特征；

（3）有一组操作，每组操作决定对象的一种行为；

（4）对象的行为可以分为两类：一类是作用于自身的行为；另一类是作用于其他对象的行为。

例如，雇员刘明是一个对象。

对象名：
　　刘明
对象的属性：
　　年龄：36
　　生日：1966.10.30
　　工资：20000
　　部门：人事部
对象的操作：
　　吃饭
　　开车

4.什么是消息？消息具有什么性质？

【解】 在面向对象程序设计中，一个对象向另一个对象发出的请求被称为“消息”。当对象接收到发向它的消息时，就调用有关的方法，执行相应的操作。例如，有一个教师对象张三和一个学生对象李四，对象李四可以发出消息，请求对象张三演示一个实验，当对象张三接收到这个消息后，确定应完成的操作并执行之。

一般情况下，我们称发送消息的对象为发送者或请求者，接收消息的对象为接收者或目标对象。对象中的联系只能通过消息传递来进行。接收对象只有在接收到消息时，才能被激活，被激活的对象会根据消息的要求完成相应的功能。

消息具有以下三个性质：

(1) 同一个对象可以接收不同形式的多个消息，作出不同的响应；

(2) 相同形式的消息可以传递给不同的对象，所作出的响应可以是不同的；

(3) 对消息的响应并不是必需的，对象可以响应消息，也可以不响应。

5.什么是抽象和封装？请举例说明

【解】 抽象是将有关事物的共性归纳、集中的过程。抽象是对复杂世界的简单表示，抽象并不打算了解全部问题，而只强调感兴趣的信息，忽略了与主题无关的信息。例如，在设计一个成绩管理程序的过程中，只关心学生的姓名、学号、成绩等，而对他的身高、体重等信息就可以忽略。而在学生健康信息管理系统中，身高、体重等信息必须抽象出来，而成绩则可以忽略。

抽象是通过特定的实例(对象)抽取共同性质后形成概念的过程。面向对象程序设计中的抽象包括两个方面：数据抽象和代码抽象(或称为行为抽象)。前者描述某类对象的属性或状态，也就是此类对象区别于彼类对象的特征物理量；后者描述了某类对象的共同行为特征或具有的共同功能。

在现实世界中，所谓封装就是把某个事物包围起来，使外界不知道该事物的具体内容。在面向对象程序设计中，封装是指把数据和实现操作的代码集中起来放在对象内部，并尽可能隐蔽对象的内部细节。

下面以一台洗衣机为例，说明对象的封装特征。首先，每一台洗衣机有一些区别于其他洗衣机的静态属性，例如出厂日期、机器编号等。另外，洗衣机上有一些按键，如“启动”“暂

6.什么是继承？请举例说明

【解】 继承所表达的是类之间的相关关系，这种关系使得某类对象可以继承另外一类对象的特征和能力。现实生活中，继承是很普遍和容易理解的。例如我们继承了我们父母的一些特征，如种族、血型、眼睛的颜色等，父母是我们所具有的属性的基础。

图 1.1 所示是一个继承的典型例子：汽车继承的层次。

以面向对象程序设计的观点，继承所表达的是类之间相关的关系。这种关系使得某一类可以继承另外一个类的特征和能力。

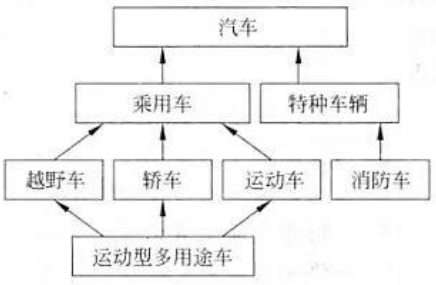

图 1.1

7.若类之间具有继承关系，则它们之间具有什么特征？

【解】 若类之间具有继承关系，则它们之间具有下列几个特性：

(1) 类间具有共享特征(包括数据和操作代码的共享)；

(2) 类间具有差别或新增部分(包括非共享的数据和操作代码)；

(3) 类间具有层次结构。

假设有两个类 A 和 B，若类 B 继承类 A，则类 B 包含了类 A 的特征(包括数据和操作)，同时也可以加入自己所特有的新特性。这时，我们称被继承类 A 为基类或父类；而称继承类 B 为类 A 的派生类或子类。同时，我们还可以说，类 B 是从类 A 中派生出来的。

8.什么是多态性？请举例说明。

【解】 面向对象系统的多态性是指不同的对象收到相同的消息时执行不同的操作。例如，有一个窗口(Window)类对象，还有一个棋子(Piece)类对象，当我们对它们发出“移动”的消息时，“移动”操作在 Window 类对象和 Piece 类对象上可以有不同的行为。

C++ 语言支持两种多态性，即编译时的多态性和运行时的多态性。编译时的多态性是通过函数重载(包括运算符重载)来实现的，运行时的多态性是通过虚函数来实现的。

9.什么是单继承，多继承？请举例说明。

【解】 从继承源上分，继承分为单继承和多继承。

单继承是指每个派生类只直接继承了一个基类的特征。图 1.2 表示了一种单继承关系。它表示 Windows 操作系统的窗口之间的继承关系。

多继承是指多个基类派生出一个派生类的继承关系。多继承的派生类直接继承了不止一个基类的特征。例如，小孩喜欢的玩具车即继承了车的一些特征，还继承了玩具的一些特征。如图 1.3 所示。

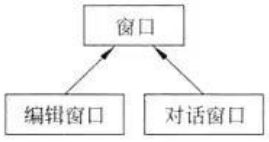

图 1.2

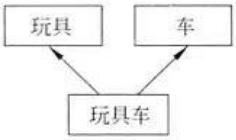

图 1.3

10.面向对象程序设计的主要优点是什么？

【解】 面向对象程序设计本质上改变了人们以往设计软件的思维方式，从而使程序设计者摆脱了具体的数据格式和过程的束缚，将精力集中于要处理对象的设计和研究上，极大地减少了软件开发的复杂性，提高了软件开发的效率。面向对象程序设计主要具有以下优点：

(1) 可提高程序的重用性；

(2) 可控制程序的复杂性；

(3) 可改善程序的可维护性；

(4) 能够更好地支持大型程序设计；

(5) 增强了计算机处理信息的范围；

(6) 能很好地适应新的硬件环境。

面向对象程序设计是目前解决软件开发面临难题的最有希望、最有前途的方法之一。

11.简述 C++的主要特点

【解】 C++ 的主要特点如下：

(1) C++ 保持与 C 的兼容，用 C 编写的软件可以用到 C++ 中；

(2) 用 C++ 编写的程序可读性好，代码结构更合理，可直接在程序中映射问题空间的结构；

(3) 生成代码的质量高；

(4) 软件的可重用性、可扩充性、可维护性和可靠性有了明显的提高，从而节省了开发费用和时间；

(5) 支持面向对象的机制，可方便地构造出模拟现实问题的实体和操作。

12.类声明的一般格式是什么？

【解】 类声明的一般格式如下：

```
class  类名{
  [private:]
    私有数据成员和成员函数
  public:
    公有数据成员和成员函数
};
```

其中：class 是声明类的关键字，类名是要声明的类的名字；后面的花括号表示类声明的范围；最后的分号表示类声明结束。

除了 private 和 public 之外，类中的成员还可以用另一个关键字 protected 来说明。这时类声明的格式可写成：

```
class  类名{
  [private:]
    私有数据成员和成员函数
  public:
    公有数据成员和成员函数
protected:
  保护数据成员和成员函数
  };
```

被 protected 说明的数据成员和成员函数称为保护成员。保护成员可以由本类的成员函数访问，也可以由本类的派生类的成员函数访问，而类外的任何访问都是非法的，即它是半隐蔽的。

13.构造函数和析构函数的主要作用是什么？它们各有什么特性？

【参考答案】构造函数可以创建对象并初始化对象，其主要特性如下。

（1）构造函数的函数名必须与类名相同，以类名为函数名的成员函数一定是类的构造函数。

（2）构造函数没有返回值类型，前面不能添加“void”。

（3）构造函数为 public 属性，否则会造成定义对象时无法调用构造函数。

（4）构造函数只在创建对象时由系统自动调用，所定义的对象在对象名后要提供构造函数所需要的实参，形式为对象名（实参表）。

（5）一个类可以拥有多个构造函数，对构造函数可以进行重载。

（6）若用户没有定义构造函数，系统会为每个类自动提供一个不带形参的构造函数。但是，此时只负责为对象的各个数据成员分配空间，而不提供初值。

析构函数的主要作用：当对象生存期结束时，负责释放对象所占的资源，其主要特征如下。

（1）析构函数也是类的特殊成员函数，其函数名与类名相同，但在类名前要加“～”号。

（2）析构函数没有返回值类型，前面不能加“void”，必须定义为公有成员函数。

（3）析构函数没有形参，不能被重载，每个类只能拥有一个析构函数。

（4）析构函数的调用也是自动执行的。

（5）在任何情况下，析构函数的调用顺序与构造函数的调用顺序正好完全相反。

（6）若用户没有定义析构函数，系统会为其提供一个默认析构函数。

14.什么是对象数组？

【解】 所谓对象数组是指每一数组元素都是对象的数组，也就是说，若一个类有若干个对象，我们把这一系列的对象用一个数组来存放。对象数组的元素是对象，不仅具有数据成员，而且还有函数成员。

15.什么是 this 指针？它的主要作用是什么？

【解】 C++ 为成员函数提供了一个名字为 this 的指针，这个指针称为自引用指针。每当创建一个对象时，系统就把 this 指针初始化为指向该对象。

每当调用一个成员函数时，系统就自动把 this 指针作为一个隐含的参数传给该函数。不同的对象调用同一个成员函数时，C++ 编译器将根据成员函数的 this 指针所指向的对象来确定应该引用哪一个对象的数据成员。

16.友元函数有什么作用？

【解】 友元函数不是当前类的成员函数，而是独立于当前类的外部函数，但它可以访问该类所有的成员，包括私有成员、保护成员和公有成员。

当一个函数需要访问多个类时，友元函数非常有用，普通的成员函数只能访问其所属的类，但是多个类的友元函数能够访问相应的所有类的数据。此外，在某些情况，例如运算符被重载时，需要用到友元函数。

17.有哪几种继承方式？每种方式的派生类对基类成员的继承性如何？

【解】类的继承方式有 public（公有继承）、protected（保护继承）和 private（私有继承）3 种，不同的继承方式导致不同访问属性的基类成员在派生类中的访问属性也有所不同。表 1-5-1 列出了基类成员在派生类中的访问属性。

表 1-5-1　基类成员在派生类中的访问属性

基类中的成员	在公有派生类中的访问属性	在私有派生类中的访问属性	在保护派生类中的访问属性
私有成员	不可直接访问	不可直接访问	不可直接访问
公有成员	公有	私有	保护
保护成员	保护	私有	保护

从表中不难归纳出以下几点：

（1）基类中的私有成员。

无论哪种继承方式，基类中的私有成员不允许派生类继承，即在派生类中是不可直接访问。

（2）基类中的公有成员。

当类的继承方式为公有继承时，基类中的所有公有成员在派生类中仍以公有成员的身份出现，在派生类内部和派生类外部都可以访问这些成员。

当类的继承方式为私有继承时，基类中的所有公有成员在派生类中都以私有成员的身份出现，在派生类内可以访问这些成员，但派生类外部不能访问它们。

当类的继承方式为保护继承时，基类中的所有公有成员在派生类中都以保护成员的身份出现，在派生类内可以访问这些成员，但派生类外部不能访问它们，而在下一层派生类内可以访问它们。

（3）基类中的保护成员。

当类的继承方式为公有继承时，基类中的所有保护成员在派生类中仍以保护成员的身份出现，在派生类内可以访问这些成员，但派生类外部不能访问它们，而在下一层派生类内可以访问它们。

当类的继承方式为私有继承时，基类中的所有保护成员在派生类中都以私有成员的身份出现，在派生类内可以访问这些成员，但派生类外部不能访问它们。

当类的继承方式为保护继承时，基类中的所有保护成员在派生类中仍以保护成员的身份出现，在派生类内可以访问这些成员，但派生类外部不能访问它们，而在下一层派生类内可以访问它们。

18.派生类能否直接访问基类的私有成员？若否，应如何实现？

【解】派生类不能直接访问基类的私有成员，但是可以通过基类提供的公有成员函数间接地访问基类的私有成员。

19.保护成员有哪些特性？保护成员以公有方式或私有方式被继承后的访问特性如何？

【解】 保护成员可以被派生类的成员函数访问，但是对于外界是隐藏起来的，外部函数不能访问它。因此，为了便于派生类的访问，可以将基类私有成员中需要提供给派生类访问的成员定义为保护成员。

C++ 规定，派生类对于保护成员的继承与公有成员的继承很相似，也分为两种情况：若为公有派生，则基类中的保护成员在派生类中也为保护成员；若为私有派生，则基类中的保护成员在派生类中成为私有成员。

20.派生类构造函数和析构函数的执行顺序是怎么样的？

【解】 通常情况下，当创建派生类对象时，首先执行基类的构造函数，随后再执行派生类的构造函数；当撤消派生类对象时，则先执行派生类的析构函数，随后再执行基类的析构函数。

21.派生类构造函数和析构函数的构造规则是怎么样的？

【解】 当基类的构造函数没有参数，或没有显式定义构造函数时，派生类可以不向基类传递参数，甚至可以不定义构造函数。

当基类含有带参数的构造函数时，派生类必须定义构造函数，以提供把参数传递给基类构造函数的途径。

在C++中，派生类构造函数的一般格式为：

```
派生类名(参数总表)：基类名(参数表)
{
    派生类新增数据成员的初始化语句
}
```

其中基类构造函数的参数，通常来源于派生类构造函数的参数总表，也可以用常数值。

在派生类中可以根据需要定义自己的析构函数，用来对派生类中增加的成员进行清理工作。基类的清理工作仍然由基类的析构函数负责。由于析构函数是不带参数的，在派生类中是否要自定义析构函数与它所属基类的析构函数无关。在执行派生类的析构函数时，系统会自动调用基类的析构函数，对基类的对象进行清理。

22.什么是多继承？多继承时，构造函数和析构函数执行顺序是怎么样的？

【解】 当一个派生类具有多个基类时，这种派生方法称为多继承。

多继承构造函数的执行顺序与单继承构造函数的执行顺序相同，也是遵循先执行基类的构造函数，再执行对象成员的构造函数，最后执行派生类构造函数的原则。在多个基类之间则严格按照派生类声明时从左到右的顺序来排列先后。而析构函数的执行顺序则刚好与构造函数的执行顺序相反。

23.在类的派生中为何要引入虚基类？虚基类构造函数的调用顺序是如何规定的？

【解】 当引用派生类的成员时，首先在派生类自身的作用域中寻找这个成员，如果没有找到，则到它的基类中寻找。如果一个派生类是从多个基类派生出来的，而这些基类又有一个共同的基类，则在这个派生类中访问这个共同的基类中的成员时，可能会产生二义性。为了解决这种二义性，C++ 引入了虚基类的概念。

虚基类的初始化与一般的多继承的初始化在语法上是一样的，但构造函数的调用顺序不同。在使用虚基类机制时应该注意以下几点：

（1）如果在虚基类中定义有带形参的构造函数，并且没有定义默认形式的构造函数，则整个继承结构中，所有直接或间接的派生类都必须在构造函数的成员初始化表中列出对虚基类构造函数的调用，以初始化在虚基类中定义的数据成员。

（2）建立一个对象时，如果这个对象中含有从虚基类继承来的成员，则虚基类的成员是由最远派生类的构造函数通过调用虚基类的构造函数进行初始化的。该派生类的其他基类对虚基类构造函数的调用都自动被忽略。

（3）若同一层次中同时包含虚基类和非虚基类，应先调用虚基类的构造函数，再调用非虚基类的构造函数，最后调用派生类构造函数。

（4）对于多个虚基类，构造函数的执行顺序仍然是先左后右，自上而下。

（5）对于非虚基类，构造函数的执行顺序仍是先左后右，自上而下。

（6）若虚基类由非虚基类派生而来，则仍然先调用基类构造函数，再调用派生类的构造函数。

24.什么是静态联编？什么是动态联编？

【解】 一个源程序经过编译、连接，成为可执行文件的过程是把可执行代码联编（或称装配）在一起的过程。其中在运行之前就完成的联编称为静态联编，又叫前期联编；而在程序运行时才完成的联编叫动态联编，也称后期联编。

静态联编是指系统在编译时就决定如何实现某一动作。静态联编要求在程序编译时就知道调用函数的全部信息。因此，这种联编类型的函数调用速度很快。效率高是静态联编的主要优点。

动态联编是指系统在运行时动态实现某一动作。采用这种联编方式，一直要到程序运行时才能确定调用哪个函数。动态联编的主要优点是：提供了更好的灵活性、问题抽象性和程序易维护性。

25.编译时的多态性与运行时的多态性有什么区别？它们的实现方法有什么不同？

【参考答案】编译时的多态性：是由同名函数根据定义时在形式参数的个数、顺序、类型方面的不同，在程序编译时就能根据实际参数与形式参数的匹配情况，确定究竟调用了哪一个函数。运行时的多态性：基类与派生类中存在的同名函数，只有在程序运行时，通过基类的指针指向基类或派生类对象，或基类的引用代表了基类或派生类的对象，确定调用的是基类还是派生类中的同名函数。编译时的多态性通过函数重载和运算符重载实现；运行时的多态性通过继承和虚函数实现。

26.简述运算重载的规则

【解】 运算符重载的规则如下：

(1) 运算符重载是针对新类型数据的实际需要，对原有运算符进行适当的改造完成的。一般来讲，重载的功能应当与原有的功能相类似。

(2) C++ 语言中只能重载原先已有定义的运算符。

(3) 不是所有的运算符都能重载，不能重载的运算符是：类属关系运算符“.”、成员指针运算符“*”、作用域分辨符“::”、sizeof 运算符和三目运算符“?:”

(4) 运算符重载不能改变运算符的操作数个数。

(5) 运算符重载不能改变运算符原有的优先级。

(6) 运算符重载不能改变运算符原有的结合特性。

(7) 运算符重载不能改变运算符对预定义类型数据的操作方式。

27.友元运算符函数和成员运算符函数有什么不同？

答：当运算符函数是一个成员函数时，最左边的操作数(或者只有左边的操作数)必须是运算符类的一个对象(或者是该类的一个引用)，this 指针作为隐含的参数传递给运算符函数。如果左边的操作数是一个不同类的对象或者是一个内部类的对象，该运算符函数必须作为一个友元函数实现，此时必须要显式的把所有的参数放在参数列表。

28.什么是虚函数？虚函数与函数重载有哪些相同点与不同点？

【解】 虚函数就是在基类中被关键字 virtual 说明，并在派生类中重新定义的函数。虚函数的作用是允许在派生类中重新定义与基类同名的函数，并且可以通过基类指针或引用来访问基类和派生类中的同名函数。

在一个派生类中重新定义基类的虚函数是函数重载的另一种形式，但它不同于一般的函数重载。当普通的函数重载时，其函数的参数或参数类型必须有所不同，函数的返回类型也可以不同。但是，当重载一个虚函数时，也就是说在派生类中重新定义虚函数时，要求函数名、返回类型、参数个数、参数的类型和顺序与基类中的虚函数原型完全相同。如果仅仅返回类型不同，其余均相同，系统会给出错误信息；若仅仅函数名相同，而参数的个数、类型或顺序不同，系统将它作为普通的函数重载，这时虚函数的特性将丢失。

29.什么是纯虚函数？什么是抽象类？

【解】 纯虚函数是一个在基类中说明的虚函数，它在该基类中没有定义，但要求在它的派生类中必须定义自己的版本，或重新说明为纯虚函数。

纯虚函数的定义形式如下：

```
virtual 函数类型 函数名(参数表)=0;
```

如果一个类至少有一个纯虚函数，那么就称该类为抽象类。

30.为什么使用模块？函数模块声明的一般形式是什么？

【解】 模板是实现代码重用机制的一种工具，它可以实现类型参数化，即把类型定义为参数，从而实现了真正的代码重用。使用模板可以大幅度地提高程序设计的效率。

函数模板的一般说明形式如下：

```
template <class 类型参数>
返回类型 函数名(模板形参表)
{
    函数体
}
```

其中，template 是声明模板的关键字，它表示声明一个模板。类型参数前需要加关键字 class(或 typename)。

31.什么是模块实参和模块函数？

【解】 将函数模板中的类型参数实例化的参数称为模板实参，函数模板对某一特定类型的实例就是模板函数。

32.什么是类模板？类模板声明的一般形式是什么？

【解】所谓类模板，实际上是建立一个通用类，其数据成员、成员函数的返回类型和形参类型不具体指定，用一个虚拟的类型来代表。使用类模板定义对象时，系统会根据实参的类型来取代类模板中虚拟的类型，从而实现了不同类的功能。

定义一个类模板与定义函数模板的格式类似，必须以关键字 template 开始，后面是尖括号括起来的模板参数，然后是类名。其具体格式如下：

```
template <typename 类型参数>
class 类名{
  类成员声明
};
```

33.函数模块与同名的非模板函数重载时，调用的顺序是怎样的？

【解】 函数模板与同名的非模板函数可以重载。在这种情况下，调用的顺序是：首先寻找一个参数完全匹配的非模板函数，如果找到了就调用它，若没有找到，则寻找函数模板，将其实例化，产生一个与之相匹配的模板函数，若找到了，就调用它。

34.C++为什么要有自己的输入输出系统？

【解】 C++ 除了完全支持 C 语言的输入输出系统外，还定义了一套面向对象的输入输出系统。为什么 C++ 还要建立自己的输入输出系统呢？这是因为在 C++ 中需要定义众多的用户自定义类型，面向对象方法的数据封装性就是通过用户所定义的类类型来体现的，而继承性和多态性也是通过对用户定义的类对象的操作来体现的。但 C 语言的输入输出系统不支持用户自定义的类型。

35.C++有哪 4 个预定义的流对象？它们分别与什么具体设备相关联？

【解】 C++ 中包含几个预定义的流对象，它们是标准输入流对象 cin、标准输出流对象 cout、非缓冲型的标准出错流对象 cerr 和缓冲型的标准出错流对象 clog。这 4 个流所关联的具体设备为：

cin	与标准输入设备相关联，通常指键盘
cout	与标准输出设备相关联，通常指显示器
cerr	与标准错误输出设备相关联（非缓冲方式），通常指显示器
clog	与标准错误输出设备相关联（缓冲方式），通常指显示器

36.cerr 和 clog 之间的区别是什么？

【解】 cerr 和 clog 之间的区别是，cerr 没有被缓冲，因而发送给它的任何内容都立即输出；相反，clog 被缓冲，只有当缓冲区满时才进行输出，也可以通过刷新流的方式强迫刷新缓冲区。

37.C++提供了哪两种控制输入输出格式的方法？

【解】 C++ 提供了两种进行格式控制的方法：一种是使用 ios 类中有关格式控制的流成员函数进行格式控制；另一种是使用称为操纵符的特殊类型的函数进行格式控制。

38.C++进行文件输入输出的基本过程是什么？

【解】 C++ 中进行文件输入输出的基本过程是，必须首先创建一个流对象，然后将这个流对象与文件相关联，即打开文件，此时才能进行读写操作，读写操作完成后再关闭这个文件。