# 第 1 章　概述

程序员的主要工作就是采用合适的程序语言，设计和编写各种各样的程序。这些程序经过翻译之后，生成计算机可以理解的目标代码，然后投入运行，并以此控制计算机的执行过程，最终完成特定的任务。程序语言是程序员和计算机进行交流的基本工具，其风格和特点直接影响着程序员的思维和解决问题的方式，以及程序的可读性。本章主要介绍计算机程序语言的发展过程，讨论结构化程序设计方法和面向对象程序设计方法各自的特点，讲述 C++语言的概貌以及 C++程序的开发过程。

## 1.1　结构化程序设计

电子计算机自从 20 世纪 40 年代诞生以来，虽然至今只走过了短短 70 年的历程，但是它对人类文明进程的贡献却是无法估量的。随着网络、多媒体等一系列相关技术的发展，计算机已经渗透到了人类生产生活中的各个领域，并成为必不可少的主要工具之一。

然而计算机毕竟是一种机器，它的功能再强大，也需要人来操纵。为了使计算机正常运行，并且按照人的意图完成各种各样的工作，就必须编写相应的程序，然后让计算机去执行它。所谓程序，就是以某种计算机语言为工具编制出来的动作序列，用于描述对某一问题的解决步骤。如同书写文章要借助于人类的自然语言一样，编写程序也需要借助于程序设计语言。人类的自然语言是人与人之间进行信息交流的工具，程序设计语言则是人与计算机之间进行信息交流的工具。计算机自一诞生就有了程序设计语言，程序设计语言的发展也十分迅速。目前新的程序设计语言如雨后春笋，层出不穷，其功能越来越强大，使用也越来越简便。我们固然可以使用不同的程序设计语言对同一个问题编写程序求解，但是解决问题的难易程度以及效果会有较大差异。事实上各种程序设计语言都有其自身的特点和规律，了解程序设计语言发展的历史过程，会有助于我们加深对程序设计语言的认识，更好地理解这些语言的规律，最终灵活地运用它编写程序，解决工程实际问题。

### 1.1.1　程序设计语言的发展

受物理元器件特性的影响，目前计算机的数制仍然是二进制，只能识别和存储由 0 和 1 组成的数字序列。机器语言就是由这些数字序列组成的，用于表示各种指令和数据。例如可以用 00000011 表示“加法”指令，用 10111111 表示“移动”指令等。计算机可以直接执行用机器语言编写的程序，而且速度很快。但是机器语言的指令非常不直观，这些由二进制数字组成的序列，对于一般人来说无异于天书，不仅难以记忆，而且容易写错。在编写机器语言程序的时候，要求程序员必须十分熟悉所操作的计算机的硬件结构，程序的通用性和可移植性较差。

到了 20 世纪 50 年代中期，为了便于理解和记忆，人们采用一些指令助记符来代替机器语言指令。例如用 ADD 表示“加法”指令，用 MOV 表示“移动”指令等。这种指令称为汇编指令，采用汇编指令的程序语言称为汇编语言，用它编写的程序称为汇编程序。但是计算机

不能直接执行汇编程序，必须翻译成机器语言程序之后才能执行。我们称前者为源程序，后者为目标程序。不同计算机的汇编指令也不尽相同，汇编语言指令与机器语言指令存在一一对应的关系。尽管汇编语言相对于机器语言来说，程序的可读性有所改善，但是编写程序时，同样需要对计算机的硬件结构比较熟悉。

汇编语言和机器语言都与计算机的硬件结构密切相关，它们统称为面向机器的语言。用这种语言编写的程序，虽然运行效率极高，但是程序员不仅需要考虑编程的思路，还需要熟悉计算机的内部结构，甚至需要涉及为数据安排具体的存储空间等诸多细节。由此导致编程的劳动强度较大，给编写大型程序以及计算机的普及推广造成了很大障碍。

为了克服面向机器的语言的这些弱点，使人们将程序设计的精力集中在求解问题上，在20世纪60年代便出现了面向过程的程序设计语言，又称为高级语言。典型的高级语言有C语言、Basic语言、Pascal语言和Fortran语言等，这些语言接近于人类的自然语言，因而较易掌握，设计出的程序也更容易理解。例如用C语言编写“计算5+3”的程序为：

```
main()
{
   int i,j,k;
      i=5;
      j=3;
      k=i+j;
   printf("k=%d\n",k);
}
```

计算机也不能直接执行高级语言程序，必须经过翻译之后才能执行。高级语言不依赖于计算机的具体硬件，其程序设计的重心在于算法和数据结构。这使得程序员可以把精力集中在解题思路和方法上，用接近于人类习惯的思维方式构思解题过程。算法一旦确定，则采用各种高级语言均可实现对同一问题的求解，因而编程的效率和质量得到了较大提高。高级语言采用结构化程序设计思想，将任务分解为一个个功能模块，并确定模块之间的调用关系，然后在程序中分别实现这些模块。上述特点使得人们可以较为容易地编写程序，完成各种复杂的功能。高级语言也因此在20世纪70年代得到了极大的发展，百花齐放，鼎盛时甚至达到了几百种之多。结构化程序设计思想也成为当时的主流程序设计思想，指导人们编写出规模越来越大的程序，以满足日益广泛而深入的应用需求。直到现在，结构化程序设计思想也仍然是一种重要的程序设计思想，在软件设计领域中发挥着它应有的作用。

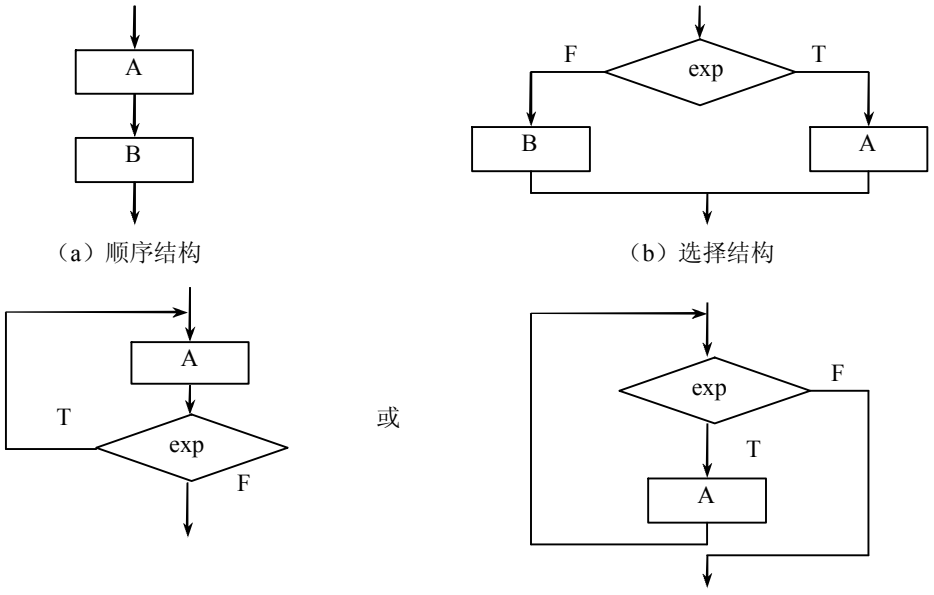
高级语言是面向过程的结构化程序设计语言，重点是描述问题求解的过程、算法和方法。问题求解的常用手段是功能分解，并把分解的结果用高级语言结构化地实现。随着程序的规模越来越大，用高级语言编写的程序逐渐暴露出数据与算法分离、代码重用困难等弱点，于是又出现了面向对象语言，例如C++语言和Java语言等。面向对象语言采用面向对象的程序设计思想，编写的程序由一个个对象组成，对象具有属性和行为，对象之间通过消息相互联系。具有相同属性和行为的对象抽象成类，并可以通过派生的方式产生新的类。面向对象语言适宜开发较大规模的软件，程序可读性强、安全性高、较易维护和扩充。面向对象语言在20世纪90年代得到了极大的发展，编写出的程序比结构化的程序更加清晰、易懂，更适宜编写大规模的程序。面向对象语言已成为当代程序设计语言的主流，面向对象的程序设计思想也成为目前软件开发中占主导地位的程序设计思想。

当前计算机程序语言的发展逐渐出现多元化、专业化的趋势。流行较广的程序语言有 Delphi、PowerBuilder 和 Visual FoxPro 等，它们一般都限定于某些特定的应用领域，或者支持某种编程特色，例如数据库开发、可视化编程等。此外，由于互联网已经进入现代信息社会，又出现了许多基于 Web 的程序语言，如 C#、ASP、JSP 和 PHP 等。总之，新的程序语言越来越易于理解和掌握，其编程环境越来越友好，功能也越来越强大。说不定将来会出现用人类的自然语言叙述问题，然后由计算机自动生成程序并执行的程序设计语言。到那时，计算机语言本身可能已经没有学习的必要了，但是程序设计思想依然会对人们设计和编写程序起着重要的指导作用。

### 1.1.2　结构化程序设计思想

思想有多远，我们就能够走多远。随着软件技术的日臻成熟，人们越来越感受到程序设计思想对于软件开发的重要意义。早期的软件设计规模较小，编写者和使用者往往是同一个人或者同一组织。这种个体化的软件开发环境使得软件设计通常是在人们头脑中进行的一个隐含的过程，人们并不太重视程序设计思想的研究。随着计算机应用的日益普及，软件的数量和规模都在急剧膨胀。人们必须及时改正程序运行中发现的错误，根据用户提出的新的需求修改程序，而运行环境的改变通常也需要修改程序。这时缺乏系统的程序设计思想的弊端就逐渐暴露出来了，许多大型软件不仅开发效率低下，而且难以维护，造成大量人力、物力资源的浪费。以上种种严重的问题统称为软件危机，人们开始认真研究解决软件危机的方法，认识到理论指导对于程序设计的重要性，从而逐步形成了一系列开发、维护软件的思想和规范，并产生了软件工程这一新兴的工程学科。

结构化程序设计的概念最早是由 E.W.Dijkstra 提出的。1966 年有人证明，只用三种基本的控制结构就能实现任意结构的算法，这三种基本控制结构就是顺序结构、选择结构和循环结构。它们的流程图如图 1-1 所示。




（a）顺序结构　（b）选择结构

（c）循环结构

图 1-1　三种基本的控制结构

（1）顺序结构。各个操作是按顺序执行的，从操作序列的第一个操作开始，按顺序执行序列的所有操作，直至序列的最后一个操作。如图 1-1（a）所示，先执行 A，然后执行 B。

（2）选择结构。对指定的条件进行判断，根据判断的结果在两条分支路径中选取其中一条执行。如图 1-1（b）所示，表达式 exp 的取值为“真”（即 T）时执行 A，为“假”（即 F）时则执行 B。

（3）循环结构。根据给定的条件是否满足，从而决定是否继续执行循环体中的操作。如图 1-1（c）所示，只要循环条件 exp 的取值为“真”（即 T），则反复执行 A。

三种基本控制结构的论断为结构化程序设计技术奠定了理论基础，并在工程实践中成功地经受了检验，软件的生产率明显得到提高。结构化程序设计技术作为一种新的程序设计思想和方法逐渐得到了人们的普遍认可，并成为程序设计的规范。它采用自顶向下逐步求精的设计方法和模块化的程序结构，并且每个模块只包含顺序、选择和循环三种控制结构。

逐步求精方法是由 Wirth 提出的一种自顶向下的设计策略，面对现实的复杂问题，首先并不触及问题解法的细节，而是应当先从全局出发，用较为自然的抽象语句来表示问题，从而得到抽象的算法；接下来对抽象算法进行细化，在这一阶段设计的算法中已经开始含有程序设计语言的成分。随着算法的不断细化，越来越多地开始实现“如何做”，算法中程序设计语言的成分也越来越多；最后当把算法全部细化为程序设计语言描述时，程序设计也就随之完成了。

结构化程序设计的重点是设计程序的功能模块，每个功能的实现是通过对数据进行一系列的加工完成的。因而其程序设计包括组织数据和加工数据两个部分，组织数据需要设计数据结构，加工数据则需要设计算法，用以描述和控制对数据结构进行加工的过程。举一个简单的例子，编写程序求圆柱体的体积。显然这个问题涉及半径、高和体积等数据，我们可以据此设计相应的数据结构；计算圆柱体的体积已有现成的数学公式，我们可以设计相应的算法。如果进一步分解任务，我们还可以设计专门计算圆柱体的底圆面积的模块。用 C 语言描述如下：

```
#include<stdio.h>
#define PI 3.1415
main()
{
    float area(float r);              /*函数声明*/
    float r,h,v;                      /*定义表示圆柱体的底圆半径、高以及体积的变量*/
    scanf("%f%f",&r,&h);              /*从键盘输入数据*/
    v=area(r)*h;                      /*计算圆柱体的体积*/
    printf("v=%6.2f\n",v);            /*输出结果*/
}
float area(float r)                   /*函数定义*/
{
    float v;
    v=PI*r*r;                         /*计算圆柱体底圆的面积*/
    return(v);
}
```

大量的实践已经证明，采用自顶向下逐步求精的方法，符合人类解决复杂问题的普遍规律，可以大大提高软件开发的效率。由于采取了先全局后局部、先整体后细节、先抽象后具体的逐

步求精方法，使得开发出来的程序层次结构清晰，因此更容易阅读和理解，方便测试与维护。

支持结构化程序设计的语言主要是高级语言，例如 Fortran、Pascal、Basic、C 等。其中最为流行的、占据主导地位的是 C 语言，支持面向对象程序设计的 C++语言即是从 C 语言脱胎而来。

## 1.2　面向对象程序设计

结构化程序设计方法既简明又实用，技术上也较为成熟，历史上曾经在大型工程计算、自动化实时控制和系统软件开发等领域取得过许多显著的成绩。随着计算机技术的不断进步，以及应用领域的进一步拓宽，软件自身的规模越来越大，结构也越来越复杂。结构化程序设计方法对缓解软件危机起到了一定的作用，但是随着软件产业的不断发展，这种作用越来越有限，并且逐渐暴露出一些问题，主要表现为以下几个方面：

（1）数据与操作分离。当数据量增大、处理过程复杂时，数据与相应操作的分离使得程序变得越来越难以理解。

（2）与人的自然思维不一致。结构化程序设计方法的设计模式与现实世界的特点并不吻合，其分析结果不能直接映射问题域，而是要经过不同程度的转化和重新组合。

（3）可重用性差。当数据结构或应用环境发生变化时，需要对程序做大量的修改。

（4）可维护性差。一旦用户需求发生变化，往往需要花费很大的代价才能使软件适应这种变化。

（5）不重视数据的安全性。在结构化程序设计方法中，未充分考虑数据的安全问题，支持这种设计方法的语言（例如 C 语言）也未提供相应的保证数据安全的机制。

用户主要关心的是程序的功能，而结构化程序设计方法是围绕实现处理功能的过程来设计程序的，因此用这种方法开发出来的软件，其结构常常是不稳定的，也就是说，用户需求的变化往往造成软件结构的较大变化。结构化程序设计方法的本质是功能分解，这种分解成子功能的过程多少带有随意性。不同的程序员在编写同一个软件时，可能经过分解而得出不同的软件结构。实际上结构化程序设计方法是与传统计算机的结构特点相吻合的，即把数据和操作分别作为独立的实体，以至于在实现时，软件已和具体的应用环境密不可分了，这正是用结构化程序设计方法开发出的软件可重用性差的内在原因。

面向对象程序设计方法与结构化程序设计方法考虑问题的角度不同，它的重点不是分解功能或者研究算法的实现，而是从系统的组成进行分解，对问题进行自然分割，以更加接近于人类思维的方式建立问题域模型。从而使得编写出的程序尽可能直接地描述现实世界，构造出模块化、可重用性好、易维护的软件，提高软件开发的效率。下面讲述面向对象程序设计方法的一些主要概念。

1. 对象

对象是客观世界中实际存在的事物。从人的认知角度来看，对象可以是有形的、可感知的事物，例如手机、汽车等；也可以是某种可理解的、无形的事物，例如一项计划、一首歌曲等。对象是构成世界的一个独立单位，它具有属于自己的静态特征和动态特征。静态特征可以用某种数据来描述，称为对象的属性；动态特征是对象所表现的行为或者所具有的功能，称为对象的行为或者方法。

在设计软件时，通常只是在问题域内考虑和识别系统中的事物，并用对象来抽象地描述它们。系统中的对象是用来表示客观事物的一个实体，它是构成系统的一个基本单位，一个对象由一组属性和对这组属性进行操作的一组行为组成。例如显示器是计算机系统的一个对象，它有尺寸、重量、价格、使用寿命以及分辨率等属性，还有显示、调整等行为。对象有如下一些基本特点：

（1）以数据为中心。数据是进行处理的主体，操作是针对对象自身数据的。

（2）对象是主动的。在结构化程序设计方法中，数据是被动地等待处理；而在面向对象程序设计方法中，是用激活对象自身的方法来对其内部的数据进行处理的。

（3）实现了封装。通常情况下外界是无法看到人的内脏的，因为它们被人体的皮肤和骨骼紧紧地包裹着。对象的数据完全被封装在对象的内部，外界无法看到和直接访问，只能通过对象提供的操作来间接地处理数据。

（4）独立性强。对象是数据和操作的集合，不同的对象各自独立地处理自身的数据，彼此之间通过发送消息完成通信。

2. 类

“王侯将相，宁有种乎？”我们可能经常听到人们说这样的话：“大家都是人，不都是只有一个脑袋，两只手吗？”实际上人们在观察客观世界时，常常会自觉地对事物进行归纳。即忽略事物的非本质特征，只注意那些与考察的目标有关的本质特征，找出一些事物的共性，并把具有共同性质的事物划分为同一个类。运用分类的方法，使得我们可以对属于某一个类的全部个体对象进行统一的描述。

在面向对象程序设计方法中，类是具有相同属性和方法的对象的集合。类与对象的关系如同零件与模具的关系，每一个对象属于一个类，对象是该类的一个实例。例如编写程序求圆柱体的体积，从面向对象程序设计的思想出发，并不急于研究具体求圆柱体体积的过程，而是先刻画圆柱体类，解决共性的问题。圆柱体类的属性有半径和高，方法有显示和计算体积等，我们可以据此定义圆柱体类。用 C++语言描述如下：

```
#include<iostream.h>
const float PI=3.1415;
class cyl
{
public:
    cyl(float x,float y);        //构造函数
    void display(void);          //显示
    float get_vol(void);         //计算圆柱体体积
    private:
    float r;                     //圆柱体底圆半径
    float h;                     //圆柱体的高
};
```

然后实现圆柱体类的方法，在程序中具体描述显示和计算体积等这些方法。用 C++语言描述如下：

```
cyl::cyl(float x,float y) //构造函数
{
    r=x;
    h=y;
```

```
}
void cyl::display(void)                    //显示
{
    cout<<"半径为"<<r<<",高为"<<h<<endl;      //显示半径和高的属性值
}
float cyl::get_vol(void)                   //计算圆柱体体积
{
    float v;
    v=PI*r*r*h;
    return(v);
}
```

在实际应用时，先创建一个圆柱体类的对象，该对象自动具有了圆柱体类所有的属性和方法；然后向它发送消息，激活该对象计算体积的方法，最终得到圆柱体的体积。用 C++语言描述如下：

```
int main()
{
    float v;                       //定义表示圆柱体体积的变量
    cyl s(1,2);                    //创建一个圆柱体对象，并进行初始化
    v=s.get_vol();                 //发送消息，计算圆柱体的体积
    cout<<"v="<<v<<endl;  //输出结果
    return(0);
}
```

3．继承

继承是面向对象程序设计方法中的一个十分重要的概念。在描述类时已经注意到，许多类具有一些共同的特征，例如轿车类和卡车类有许多共同的属性和方法。我们可以把这些共同特征提取出来，形成一个汽车类，而轿车类和卡车类则是从汽车类派生出来的，这就是继承。

继承在编写程序时具有重要的意义。我们能够利用继承来描述类之间的相似性，其他的类可以方便地继承这些相似性，避免了重复定义和开发，大大提高了程序的可重用性。类的继承性使得编写出的软件具有开放性、可扩充性，减轻了工作负担。继承还提供了规范的类的层次结构，使得公共特性能够得到共享，提高了软件开发的效率。

4．多态性

多态性是指不同类的对象在接收到同一个消息后，做出的响应各不相同。例如在几何图形类中定义了一个绘图方法，显然由于不知道具体的图形种类，是无法描述绘图的具体方法的。圆类和多边形类从几何图形类继承而来，都具有绘图的方法，但其功能却各不相同。进一步地从多边形类派生出三角形类和矩形类，它们同样拥有绘图这一方法，而功能又各不相同。这样当用户想绘制几何图形时，只需要发送同样的消息给这些不同类的对象，圆类的对象在收到消息后画出一个圆，三角形类的对象在收到消息后画出一个三角形，而矩形类的对象在收到消息后画出一个矩形。多态性机制不仅增加了软件的灵活性，进一步减少了信息冗余，方便了用户的操作，而且显著地提高了程序的可重用性和可扩充性。

综上所述，面向对象程序设计方法有着诸多优点，以对象为核心，强调对现实世界的模拟而不是处理的过程，通过对象把数据和相关操作结合在一起，构成一个整体，更加易于理解，增加了程序的可读性。引入类、继承以及多态性等机制，大大增强了程序的可重用性，降低了

构建大规模软件的难度。面向对象程序设计方法以对象为实体，而不是以功能为实体，软件的稳定性较好，容易测试和调试，可维护性强。面向对象程序设计方法虽然并不一定能够缩短软件的开发周期，但是从长远角度来看，它有效地改进了软件的质量。与结构化程序设计方法相比，面向对象程序设计方法更适于编写规模较大的程序。

## 1.3 C++语言介绍

伟大的C++语言之父，Bjarne Stroustrup博士曾经说过：“一种程序设计思想要为人所用，不仅语言的特性必须是典雅的，而且它必须在真正的程序环境中能经得起考验。”面向对象程序设计方法的提出，以及它在编写大规模程序方面显示出的优越性，使人们开始重视面向对象程序设计语言的研究。在面向过程的ALGOL、Ada和Modula-2等语言的基础上，逐步演变形成了面向对象的程序设计语言。20世纪60年代美国国防部投入巨大的人力和物力，研制开发了Ada语言。Ada语言并非面向对象的程序设计语言，但它具有的模块化、信息隐藏、数据抽象和并发执行等特点，对于面向对象程序设计方法和技术起到了积极的推动作用。人们普遍认为，Ada语言是一种基于对象的程序设计语言。

1967年出现了Simula 67语言，它是面向对象程序设计语言的鼻祖，提出了对象的概念，并且支持类和继承。随后出现的Smalltalk语言继续丰富和发展了面向对象程序设计的概念，并且提供了更加严格的信息隐藏机制。1980年问世的Smalltalk-80语言是Smalltalk语言的改进版，开始向世人展现面向对象程序设计的魅力。1982年美国AT&T公司贝尔实验室的Bjarne Stroustrup博士在C语言的基础上引入并扩充了面向对象的概念，发明了一种新的程序语言。为了表达该语言与C语言的渊源关系，它被命名为C++。此后C++语言不断地完善，1990年C++语言引入模板和异常处理的概念，1993年引入运行时类型识别（RTTI）和名字空间（Name Space）的概念。1997年C++语言成为美国国家标准（ANSI），1998年C++语言又成为了国际标准（ISO），目前C++语言已成为使用最广泛的面向对象程序设计语言之一。

总的来说，面向对象程序设计语言可以分为两大类：一类是纯面向对象语言，例如Smalltalk、Eiffel和Java等；另一类是混合型面向对象语言，一般是在结构化程序设计语言的基础上增加了面向对象的机制，例如C++等语言。纯面向对象语言着重支持面向对象方法的研究，而混合型面向对象语言着重提高运行的效率以及提供强大的功能。C++语言是以C语言为基础的，支持C语言的所有语法和几乎所有技术，因此也有人把C++语言看作是C语言的超集。同时C++语言支持面向对象程序设计方法的所有概念，它是一种非常实用的、功能极为强大的程序语言，相对而言较难掌握。

用C语言编写的程序通常能被C++语言完全接受，那么与C语言相比，C++语言又有哪些不同呢？此处列举以下几点，仅供参考：

（1）C++提供了class和virtual等语法，全面支持面向对象程序设计方法。

（2）C++是强类型语言。C++语言在编译阶段就对程序进行了严格的类型检查，尽可能地在早期向程序员报告程序的错误。

（3）C++提供了friend和const等语法，既保证了数据的安全性，又为程序员提供了提高访问效率的方法。

（4）C++提供了函数重载、运算符重载以及模板等机制，使程序的可读性和编写效率得

以提高。

（5）C++通过输入输出流类的对象完成数据的输入和输出，功能更为强大，编程实现更为容易。

（6）C++引入类作用域和名字空间的概念，大大拓宽了程序的作用域。

（7）C++为动态内存分配提供了 new 和 delete 运算符，方便了动态内存分配和回收的编程。

（8）C++引入引用的概念，既保证了程序运行的效率，又兼顾了程序的可读性。

（9）C++提供异常处理的机制，大大增强了程序的可靠性和健壮性。

（10）C++提供标准字符串类，使得在程序中处理字符串变得十分容易。

（11）C++提供标准容器类，使得在程序中处理群体数据较为方便。

总之，C++语言全面超越了 C 语言，功能更为强大，程序设计方法也更为先进。可以毫不夸张地说，C++语言在所有的程序语言中，语法最为复杂，功能最为强大，堪称当之无愧的程序语言之王。

下面介绍一个最简单的 C++程序，以使读者对 C++语言有一些感性认识。

**【例 1.1】**简单的 C++程序。

```
#include <iostream.h>
 int main()        //主函数
{
   cout<<"Hello!"<<endl;
   cout<<"Welcome to C++!"<<endl;
   return(0);
}
```

运行情况如下：

```
Hello!
Welcome to C++!
```

**说明：**

（1）我们可以发现 C++程序与 C 程序在很多地方是相同的。例如都有一个 main 函数，用来控制程序执行的开始和结束。函数不仅是 C 程序的基本组成单位，也是 C++程序的重要组成部分。对象的行为是用函数描述的，C++程序本质上也是以函数为驱动的。函数由函数头部和函数体组成，函数头部描述了该函数的名称、返回值类型以及参数的个数和类型等信息，函数体由一对花括号包围，描述函数具体的动作和功能。

（2）//是 C++程序的注释符，表示从//开始直到本行结尾的字符序列都是程序的注释，不进行编译。与 C 程序的注释符/*…*/不同的是，它不能跨行。

（3）与 C 程序相同，C++程序一般也是在函数体中书写语句，并以分号作为每一条语句的结束。显然在这个简单的 C++程序中，出现了 3 条语句。

（4）cout 是标准输出流类的对象，与插入运算符“<<”配合，输出字符串等数据，字符串以一对双引号作为标识。C 程序调用输入输出库函数，需要包含头文件 stdio.h；C++程序使用输出流类的对象，则需要包含头文件 iostream.h。

（5）endl 是 C++的输入输出流操纵符，与 C 程序的转义字符'\n'的功能相同，它也表示回车换行。

（6）语句 return(0);表示 main 函数运行完毕，实际上该程序也运行完毕；返回系统，并

且函数的返回值为 0，表示正常结束。

我们将在以后各章详细地介绍 C++语言的语法以及 C++程序的编写方法。

## 1.4 C++程序的开发环境

由于计算机只能识别二进制代码，C++程序是无法在计算机上直接执行的。因此要想编写一个完整的 C++程序并完成运行，必须在 C++程序的开发环境中，把编写好的 C++语言源程序经过编辑、编译和链接等步骤后，才能形成可执行的程序。图 1-2 表示了这一过程。

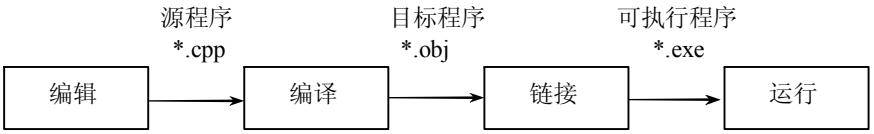



图 1-2 程序执行步骤

目前较为流行的 C++程序开发工具有 Visual C++ 6.0、Borland C++ Builder 以及 Visual Studio.NET 等。工欲善其事，必先利其器，下面以 Visual C++开发环境为例，介绍 C++程序的开发过程。Visual C++ 6.0 是著名的 Microsoft 公司推出的、使用极为广泛的可视化程序开发工具，它提供了基于 Windows 平台的 C++语言集成开发环境。Visual C++ 6.0 由于其功能强大、灵活性好、扩展性强等优点，在各种 C++语言开发工具中脱颖而出。它不仅可以用来开发普通的 C++程序，还可以用来开发各种各样的 Windows 应用程序。

成功安装了 Visual C++ 6.0 软件之后，可以用多种方法启动。例如单击“开始”→“程序”→Microsoft Visual Studio 6.0→Microsoft Visual C++ 6.0 命令，即可启动 Visual C++，进入 Microsoft Visual C++ 6.0 的集成开发环境。

如图 1-3 所示，Visual C++ 6.0 的主窗口由标题栏、菜单栏、工具栏、工作区窗口、编辑区窗口、输出窗口和状态栏组成。最上端的标题栏显示应用程序名和打开的文件名，菜单栏和工具栏下面是工作区窗口和编辑区窗口，再往下依次是输出窗口和状态栏。

（1）工作区窗口。列出了应用程序的一些重要信息，如类、工程文件、资源等。在工作区窗口中的任何标题或图标处右击，都会弹出快捷菜单，其中包含了与当前状态有关的一些常用操作。

（2）编辑区窗口。对源程序代码和工程资源（包括对话框资源、菜单资源等）进行设计和处理，该窗口能够一一显示各种 C++源程序的代码文件、资源文件和文档文件等。

（3）输出窗口。显示编译、调试和查询的结果，并给出必要的提示，帮助用户修改程序开发中出现的各种错误。

（4）状态栏。显示当前操作或所选命令的提示信息。

在 Visual C++ 6.0 环境中，创建一个普通的 C++程序的步骤是：先用 AppWizard 创建 C++控制台应用程序的框架，在编辑区窗口编写源程序，然后编译、链接，最后执行。AppWizard 能够帮助程序员快速创建一些常用的应用程序框架，例如普通的 C++应用程序即控制台应用程序、Windows 应用程序、DLL 程序等。

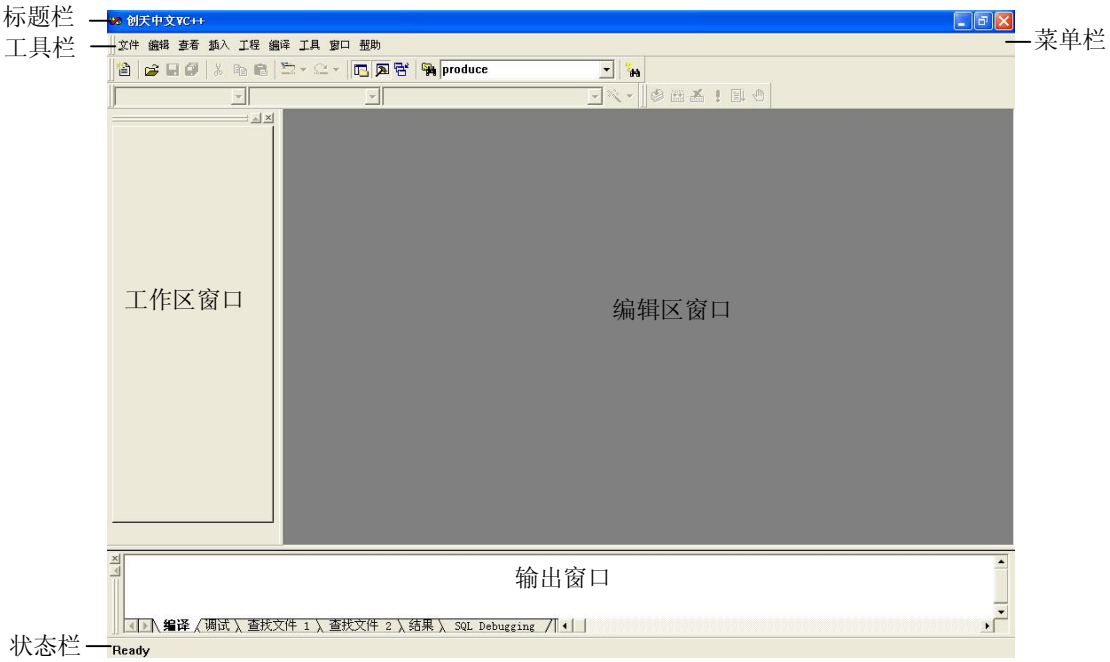



图 1-3　Visual C++ 6.0 开发环境

在编写 C++程序之前，需要先创建一个工程。单击“文件”→“新建”命令，在弹出的“新建”对话框中选择“工程”选项卡，如图 1-4 所示。

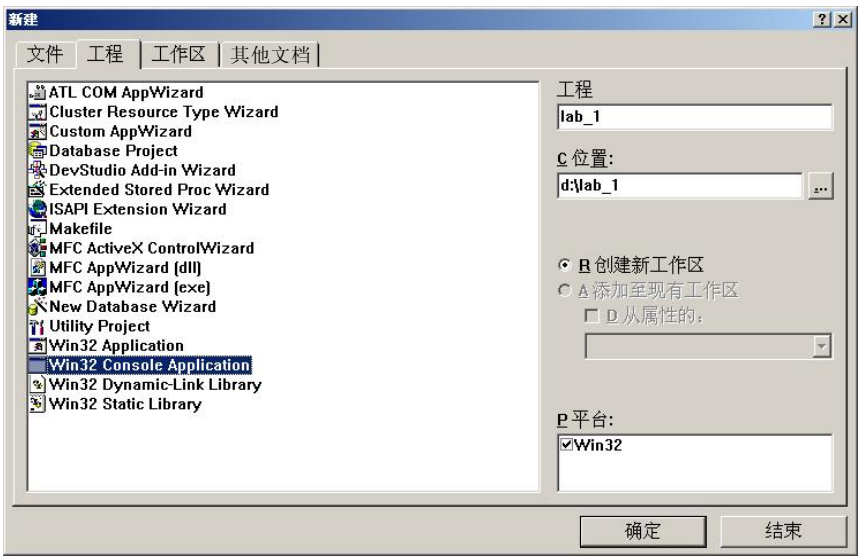



图 1-4　创建 C++程序的“工程”选项卡

在“工程”选项卡中选择 Win32 Console Application（Win32 控制台应用程序）。先在“工程”文本框中输入工程的名字，之后在“位置”文本框中指定工程的路径，单击“确定”按钮。然后按照提示一步步做下去，最终创建一个新的工程。Visual C++ 6.0 会为该工程新建一个文件夹，存放所有和 C++程序有关的文件，并由工程进行统一管理。

接下来创建 C++源程序文件。单击“文件”→“新建”命令，在弹出的“新建”对话框中选择“文件”选项卡，如图 1-5 所示。

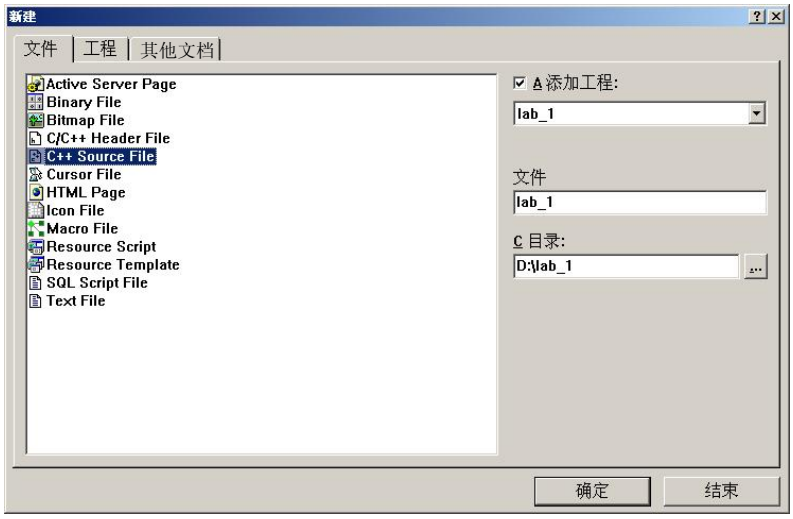



图 1-5　建立 C++源程序文件

在文件列表中选择 C++ Source File，并在“文件”文本框中输入文件的名字，单击“确定”按钮，完成 C++源程序文件的创建。随后自动出现编辑区窗口，程序员就可以在其中编写 C++程序，如图 1-6 所示。

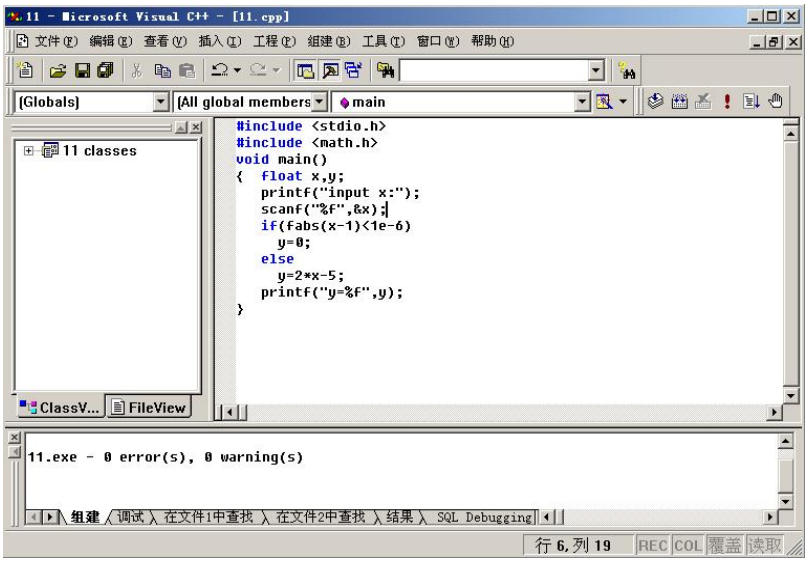



图 1-6　编写 C++程序

C++程序编写完毕之后，就进入编译、运行阶段。单击“编辑”→“编译”和“构建”命令，即可对 C++程序分别进行编译和链接。如果出现语法错误或者其他错误，系统就会在输出窗口中显示错误提示信息，程序员可以据此修改程序。

如果程序编写无误，则可以单击“编辑”→“执行”命令来运行 C++程序。这时会出现

运行结果显示窗口，程序员能够通过该窗口观察程序的运行情况，如图 1-7 所示。


图 1-7　程序运行结果显示窗口

## 1.5　小结

本章主要介绍了面向对象程序设计方法和 C++语言的基本情况。面向对象程序设计是一种较为先进的编程思想，它的要素有类、对象、继承和多态性等。对象是程序的基本组成单位，对象有属性和方法，两者密切结合，并把信息封装在对象的内部。拥有一些相同特征的对象可以抽象成类，对象是类的一个实例，对象之间通过消息进行通信。通过继承，一个类可以从另一个类自动获得大量属性和方法；利用多态性，可以实现调用界面的统一性，提高软件的灵活性。

C++语言是在 C 语言的基础上发展起来的，它是一种混合型的程序语言，能够完全兼容 C 程序。C++语言全面支持面向对象程序设计方法，并提供了许多自动化机制，能够迅速地开发较大规模的程序。

## 习题一

1．简述计算机程序设计语言的发展过程。
2．简述面向对象程序设计方法的思想。
3．C++语言与 C 语言的区别有哪些？
4．列举几种常见的 C++语言开发工具。
5．举例说明 C++程序的开发过程。
6．编写一个简单的程序，能够在屏幕上显示以下内容：

```
**********************************
**********************************
            Hello
**********************************
**********************************
```