第 3 章　程序设计语言

本章主要从程序的概念与表示和程序设计语言的发展介绍程序设计语言，并通过选择程序设计语言引入 C 语言，简述 C 语言的特点及 C 语言的编程规范。第一部分首先从解决生活中的程序入手讲解程序概念，用计算机中的程序解决实际问题，并通过绘制流程图明晰程序的执行步骤；第二部分简述了程序设计语言的发展历程，并将程序设计语言分类；第三部分主要讲解 C 语言的特点及编写 C 语言程序时需要注意的编程规范。

3.1　程序的概念与表示

程序在我们生活中的定义是办事情的章程，在计算机中的定义是执行某个任务所要经历的一系列操作，本节会从生活中的程序引入，让我们更好地理解计算机中程序的概念与表示。

3.1.1　生活中的程序

什么是程序呢？我们先来看看生活中的程序。

ATM 机还没有普及的时候，每个人都有拿着存折储蓄卡去银行取钱的经历。整个过程大致可以分为如图 3-1 所示的 6 个环节。

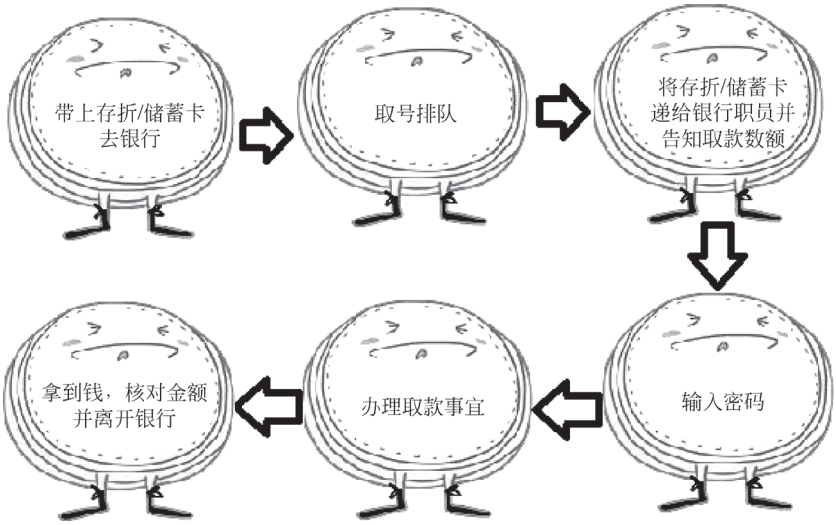

图 3-1　银行取钱过程

这是一个生活中很常见的办理事宜的程序，那如何把日常生活中取钱的例子通过计算机表示出来呢？什么是计算机的程序？如何编写计算机程序？这些，都将在这一章中进行讲解。

我们来看一个问题：如何将东西装进冰箱？

完成这件事需要三步，分别是打开冰箱→将东西放进冰箱→关上冰箱。这就是最简单的顺序执行的程序。

程序是什么？通俗地说，程序可以指连贯的活动、作业、步骤、决断、计算和工序，当它们依照严格规定的顺序发生时即可实现特定目标或解决特定问题。

C 语言程序是一种结构化的程序，那么什么是结构化的程序呢？结构化程序就是首先将一个复杂的问题分解成互相独立的几个部分（合理抽象），然后每个独立部分可以通过简单的语句或结构来实现，这种分解问题的过程就是算法设计的过程（高效计算）。

3.1.2　计算机中的程序

计算机的产生解决了很多实际问题，而这些问题的解决离不开计算机中的程序，用计算机解决问题之前，首先要把解决步骤描述出来。

来看下面的例子。

假如要求从键盘输入 3 个数，找出其中最小的那个数，将其输出在屏幕上，请给出解决这个问题的算法。

分析：程序对于从键盘输入的 3 个数必须用 3 个变量（暂时可理解为一个空盒子，用于存放数据）来保存，a，b，c 代表输入的 3 个数，另外，还需要一个变量 min 来保存最小的那个数。

（1）先比较 a 和 b 的值，把数值小的放入 min 中。

（2）再将 min 与 c 比较，再把数值小的放入 min 中。

（3）经过两次比较，min 中已存放的是 a，b，c 这 3 个数中最小的数， 把 min 的值输出就是所需结果。

上面的思考过程很重要，图 3-2 是上题内容形象化的展示。

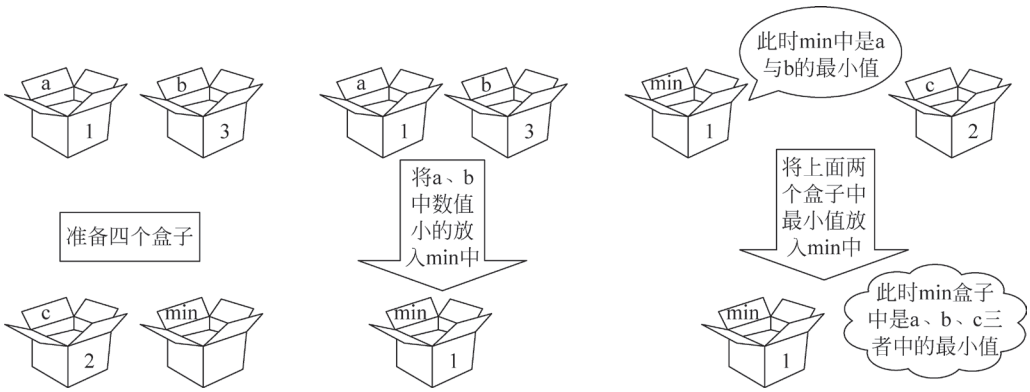

图 3-2　算法分析

根据图 3-2 的算法分析，我们可以将比较三个数中最小值的程序，从自然语言转化为以下的算法步骤并加以改进。

算法步骤：

(1) 输入3个数，其值分别赋给3个变量a，b，c；

(2) 把a与b中较小的那个数放入变量min中；

(3) 把c与min中较小的那个数放入变量min中；

(4) 输出最后结果min的值。

改进版：

(1) 输入3个数，其值分别赋给3个变量a，b，c；

(2) 比较a与b的值，如果a<b，则min=a，否则min=b；

(3) 比较c与min的值，如果c<min，则min=c；

(4) 输出最后结果min的值。

注意：上述中min = a，是将a中的数放入min中。

通过算法描述的步骤，可以很方便地用程序语言来实现。

正如前一个案例分析的一样，不是所有的程序或者流程都是顺序执行的，很多时候需要根据情况复杂度来判断，做出选择并处理。

ATM取款机的工作流程为：用户插入银行卡→输入密码→判断密码是否正确→控制密码输入次数→取款等业务→退卡。

模拟ATM取款操作的过程如图3-3所示。

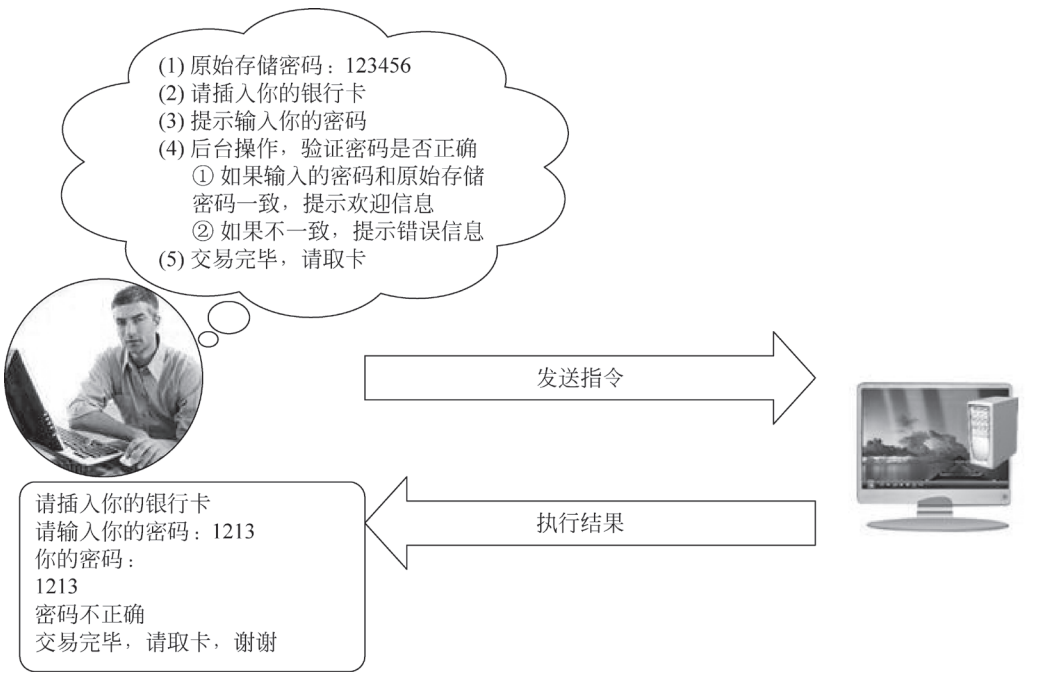

图3-3　ATM取款过程

通过上述一系列案例的描述，你是否对如何将日常生活中的程序映射到计算机程序有一定的了解了呢?

计算机程序是为实现特定目标或解决特定问题而用计算机语言编写的命令序列的集合，是告诉计算机应如何完成一个任务的。

程序由一系列指令组成，指令是指示计算机做某种运算的命令，通常包括以下几类。

（1）输入：从键盘、文件或者其他设备获取数据。

（2）输出：把数据显示到屏幕，或者存入一个文件，或者发送到其他设备。

（3）基本运算：执行最基本的数学运算和数据存取。

（4）测试和分支：测试某个条件，然后根据不同的测试结果执行不同的后续指令。

（5）循环：重复执行一系列操作。

通常在设计计算机程序之前，需要先理清程序的流程并达成共识，那么如何才能更好地理清程序的流程呢?

3.1.3　流程图

前面说程序是指连贯的活动、作业、步骤、决断、计算和工序，当它们依照严格规定的顺序发生时即可实现特定目标或解决特定问题，那如何直观地描述一个程序呢?

回到最开始讲的 ATM 取款机的工作流程，用户插入银行卡→输入密码→判断密码是否正确→控制密码输入次数→取款等业务→退卡，你能看出这里隐含了一个循环判断吗? 就是当密码输入错误在限制的次数内可以再次输入密码，显然通过文字的描述不够直观，如图 3-4 所示，每一步都一目了然，这就是描述程序的工具——流程图。

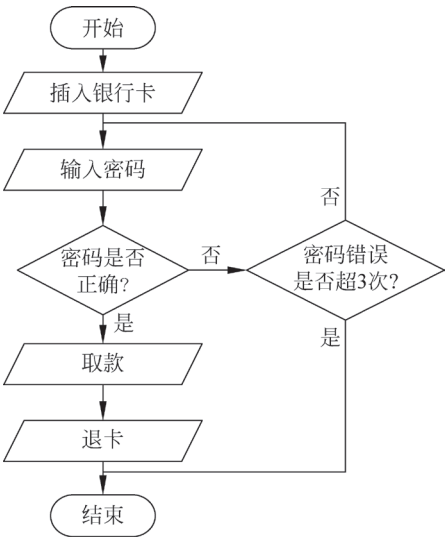

图 3-4　ATM 取款机工作流程

算法流程图的符号采用美国国家标准化协会(ANSI)规定的一些常用的流程符号，这些符号和它们所代表的功能含义如表 3-1 所示。

表 3-1　常用的算法流程图符号和功能含义

流程图符号	名　　称	功 能 含 义
	开始 / 结束框	代表算法的开始或结束。每个独立的算法只有一对开始 / 结束框
	数据框	代表算法中数据的输入或数据的输出
	处理框	代表算法中的指令或指令序列。通常为程序的表达式语句，对数据进行处理
	判断框	代表算法中的分支情况，判断条件只有满足和不满足两种情况
	连接符	当流程图在一个页面画不完的时候，用它来表示对应的连接处。用中间带数字的小圆圈表示，如①
	流程线	代表算法中处理流程的走向，连接上面各图形框，用实心箭头

一般而言，描述程序算法的流程图完全可以用表 3-1 中的 6 种流程图符号来表示，通过流程线将各框图连接起来，这些框图和流程线的有序组合就可以构成众多不同的算法描述。

对于结构化的程序，表3-1所示的6种符号组成的流程图值包含3种结构：顺序结构、分支结构和循环结构，一个完整的算法可以通过这3种基本结构的有机组合来表示。掌握了这3种基本结构的流程图的画法，就可以画出整个算法的流程图。

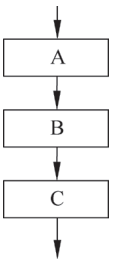

图 3-5　顺序结构的流程

1. 顺序结构

顺序结构是一种简单的线性结构，由处理框和箭头组成，根据流程线所示的方向，按顺序执行各矩形框的指令。流程图的基本结构如图3-5所示。

指令A、指令B、指令C可以是一条指令语句，也可以是多条指令，顺序结构从上到下依次执行A，B，C。

2. 选择 / 分支结构

选择 / 分支结构由判断框、处理框和箭头组成，先要对给定的条件进行判断，看是否都满足给定的条件，根据条件结构的真假而分别执行不同的处理框，其流程图的基本形式有两种，如图3-6所示。

图3-6（a）所示情况的执行顺序为：先判断条件，当条件为真时，执行A，否则执行B。

图3-6（b）所示情况的执行顺序为：先判断条件，当条件为真时，执行A，否则什么也不执行。

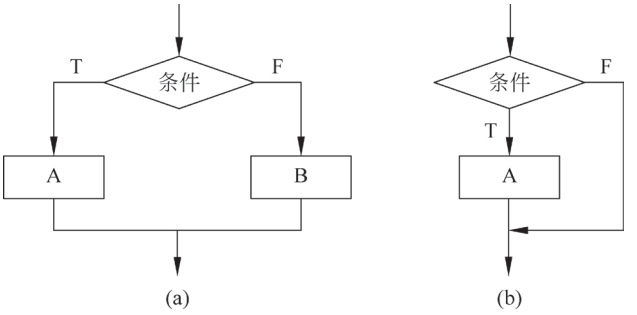

图 3-6　选择 / 分支结构的流程

3. 循环结构

同选择 / 分支结构一样，循环结构也是由判断框、处理框和箭头组成的。但循环结构是在某个条件为真的情况下，重复执行某个框中的内容。图3-7为while循环的流程。

任何两个按顺序放置的处理框可以合并为一个处理框来表示（执行框如图3-6、图3-7表示的一个基本结构）。一个完整的结构化的流程图经过多次转化后，最终都是可以转化为如图3-8所示的最简形式。

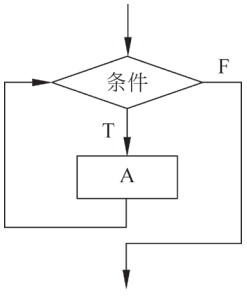

图 3-7　while 循环流程

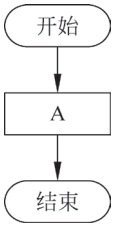

图 3-8　结构化程序的最简形式

3.2　程序设计语言发展简述

对于计算机而言，要编写程序就必须使用计算机语言，计算机语言是指编写程序时，根据事先定义的规则而写出的预定语句的集合，计算机语言经过多年的发展已经从机器语言演化到高级语言。

3.2.1　软件的产生

通过前面的学习，我们知道计算机是由硬件系统和软件系统组成。其中软件是由程序设计语言实现的，例如 C、Java 等这些都是我们所熟知的程序设计语言。从层次关系的视角来看，通过底层的计算机硬件，搭建操作系统和编译程序，然后利用我们所学的程序设计语言进行编写、编译程序，最终得到我们所需要的应用程序，就是所谓的软件，例如我们用的 IE、QQ 等。

3.2.2　程序设计语言发展史

从发展历程来看，程序设计语言可以分为前计算机时代、机器语言时代、汇编语言时代和高级语言时代。

1. 前计算机时代

19 世纪的欧洲，数学在社会各个领域应用越来越广，处处都有数学方程式的计算。那个时候计算尺、简易计算器相继发明了，可是还是不能满足大多数计算的要求，因为它们只能做 1 次运算，要算 1 个方程组，应该是好几次运算的 1 个序列，那怎么办呢？究竟需要什么样的计算机器呢？这时差分机出现了，它是 1822 年，由 Babbage 在前人计算尺和计算器的基础上发明的，它能够按照设计者的意思，自动处理加减乘除、乘方、对数运算；Babbage 提出了控制中心和存储程序思想，并且程序可以通过选择进行跳转；设计语言类似于今天的汇编语言。但它也存在着缺点：速度太慢；功能集中于几类数学运算，不全面，不能算通用的计算机。

20 世纪 40 年代，第二次世界大战全面爆发，各个参战国对新武器的狂热度不断上升，于是，新一代的导弹技术成了需要迫切的热门技术，但是，想要制作更加精良的导弹，需要精确控制导弹的弹道，这时候，需要计算一大堆的微分方程组，怎样才能更好更快地计算这些方程组呢？这便迎来了机器语言时代。

2. 机器语言时代

为了解决战争对新武器的需要，1946 年，第一台通用计算机 ENIAC 问世。ENIAC 比当时已有的计算装置要快 1000 倍，而且还有按事先编好的程序自动执行所有计算的功能（算术运算、逻辑运算、循环处理和存储数据）。但是，ENIAC 在编制计算语言上并没有使用类似汇编的语言，而是使用了 01 串的机器语言。它的缺点是：程序员必须手动控制计算机，没有存储程序，完成一个运算过程需要搭建一次电子线路；ENIAC 采用十进制数据存储和表达方式，不利于计算机部件的生产；使用机器语言，非专业人员不能编写和阅读。同样出于战争对新武器的需要，1945 年，冯・诺依曼和他的研究小组在共同讨论的基础上，发表了一个全新的“存储程序通用电子计算机方案”——EDVAC。EDVAC 解决了 ENIAC 的第一个问题：程序不能存储问题。EDVAC 两大设计

思想：二进制，大大有利于计算机部件的生产；EDVAC 方案明确奠定了新机器由五个部分组成，包括：运算器、逻辑控制装置、存储器、输入和输出设备（冯·诺依曼体系的提出），形成了至今为止一直使用的冯·诺依曼体系结构，并描述了这五部分的职能和相互关系，而且程序是存储在存储器中。但它也存在着缺点：仍然使用01串的机器语言，想要编制计算机程序需要有相当的专业水平。

讲了这么多，机器语言到底是如何编程呢？我们看下面的例子。

假设某型号计算机共有 16 个寄存器，用 4 位进行编号；共支持 256 个长度为 1B 的内存单元，用8位进行编号；每条指令占2B，前4位是操作的编码，后12位是操作的参数。假设有如表 3-2 所示的几条指令：

表 3-2　操作码对应的操作数和描述

操作码	操作数	描　　述
0000	XY	将数值 XY（8 位）放到编号 R（4 位）的寄存器中
0001	ST	将寄存器 R（4 位）和 S（4 位）的内容相加放到 T（4 位）中
0010	XY	将寄存器 R（4 位）的内容写到编号为 XY（8 位）的内存单元中

则求解 3+7 并把结果写在 4 号内存单元的机器语言程序如下：

0000 0001 0000 0011　0 号寄存器的值置为 3；

0000 0010 0000 0111　1 号寄存器的值置为 7；

0001 0000 0001 0010　0 号和 1 号寄存器的值相加，结果放在 2 号寄存器中；

0010 0010 0000 0100　2 号寄存器的值写到内存 4 号单元中。

3. 汇编语言时代

机器语言由 01 串的数字组成，程序极其难懂，非专业人员不能编写，而且极易出错。汇编语言（assembly language）是面向机器的程序设计语言。在汇编语言中，用助记符（mnemonic）代替操作码，用地址符号（symbol）或标号（label）代替地址码。这样用符号代替机器语言的二进制码，就把机器语言变成了汇编语言。所以汇编语言亦称为符号语言。使用汇编语言编写的程序，机器不能直接识别，要由一种程序将汇编语言翻译成机器语言，这种起翻译作用的程序叫汇编程序，汇编程序是系统软件中语言处理的系统软件。汇编程序把汇编语言翻译成机器语言的过程称为汇编。说简单点，就是程序员编写汇编指令通过编译器编译成计算机能懂的机器码。1949 年，从美国回到剑桥大学的威尔克斯在仔细研究了 EDVAC 之后，在 EDVAC 为蓝本的基础上，研制了 EDSAC，在 EDSAC 中，汇编语言的雏形开始出现。汇编语言的本质：用助记符代替机器指令的操作码，用地址符号或标号代替指令或操作数的地址。

接下来我们看一个汇编语言编程的例子，用计算机设计汇编语言。假设用 mov（move，移动）表示 0000 指令，用 add（相加）表示 0001 指令，用 str（store，存储）表示 0010 指令，则求解 3+7 并把结果写入 4 号内存单元中。相应的汇编语言程序如下：

```
mov 0,3
mov 1,7
add 0 1,2
str 2,4
```

计算机设计汇编语言的优点：相比机器语言，程序变得容易理解，容易编写；缺点：不同芯片的汇编语言不同，因此程序不能通用。

4. 高级语言时代

高级语言时代又分为三个时代：前结构化程序设计语言时代、结构化程序设计语言时代和面向对象程序设计时代。

1）前结构化程序设计语言时代

为了解决汇编语言不能各个平台通用的缺点，1951 年，美国 IBM 公司约翰·贝克斯（John Backus）着手研究开发 FORTRAN 语言，这也是世界上第一种高级语言。FORTRAN，是由 FORmula TRANslator 两个单词前几个字母拼成的，意思是公式翻译语言。它有以下两个优点。

（1）平台无关性，比较适合科学计算领域。

（2）积累了很多程序，有丰富的程序库。

FORTRAN 语言出现之后，程序的编写迅速推广到各个需要数学计算的领域，于是出现越来越多的新算法，从而推动了算法领域的研究。随着对算法研究的越来越深入，一门纯面向算法的语言呼之欲出，1958 年，由 Edsger Wybe Dijkstra 设计的 ALGOL 语言产生了。ALGOL 语言的优点如下。

（1）引进局部性概念、动态、递归、巴克斯瑙尔范式 BNF（Backus-Naur Form），大大增加了编程的灵活性。

（2）为软件自动化及软件可靠性的发展奠定了基础。

各种语言相继出现，计算机程序的使用范围也越来越广，但是缺乏一门便于初学者学习的语言。1964 年，美国达特茅斯学院的两个教员约翰·基米尼（J. Kemeny）和托马斯·卡茨（T.Kurtz）开发了 BASIC 语言。BASIC 语言的优点如下。

（1）简单易学。

（2）这种语言只有 26 个变量名，17 条语句，12 个函数和 3 个命令，这门语言叫作“初学者通用符号指令代码”——Beginners，All-purpose Symbolic Instruction Code。

2）结构化程序设计语言时代

20 世纪 60 年代，随着任务的逐渐复杂，早期的命令式程序设计语言变得越来越庞大，程序变得杂乱无章，难以阅读和维护，质量严重下降，编程代价巨大。为了解决这个问题，计算机科学家们提出了结构化程序设计的概念。

结构化程序设计（structured programming）是进行以模块功能和处理过程设计为主的详细设计的基本原则。其概念最早由 E.W.Dijkstra 于 1965 年提出，是软件发展的一个重要的里程碑。他的主要观点是采用自顶向下、逐步求精及模块化的程序设计方法；使用三种基本控制结构构造程序，任何程序都可由顺序、选择、循环三种基本控制结构构造。结构化程序设计主要强调的是程序的易读性。

结构化程序设计的要点如下。

（1）主张使用顺序、选择、循环三种基本结构来嵌套联结成具有复杂层次的结构化程序，严格控制 GOTO 语句的使用。

（2）“自顶而下，逐步求精”的设计思想，其出发点是从问题的总体目标开始，抽象低层的细节，先专心构造高层的结构，然后再一层一层地分解和细化。

（3）“独立功能，单出、入口”的模块结构，减少模块的相互联系，使模块可作为插件或积木使用，降低程序的复杂性，提高可靠性。

接下来来了解有哪些结构化语言。为了实现一个良好的结构化程序设计语言，并用于教学中，瑞士苏黎世联邦工业大学的 Niklaus Wirth 教授于 20 世纪 60 年代末设计并创编了 Pascal 语言。IOI（国际奥林匹克信息学竞赛）把 Pascal 语言作为三种程序设计语言之一。Pascal 语言的优点如下。

（1）严格的结构化形式。

（2）丰富完备的数据类型。

（3）运行效率高。

（4）查错能力强，可以被方便地用于描述各种算法与数据结构，有益于培养良好的程序设计风格和习惯。

1965 年时，贝尔实验室加入一项由通用电气和麻省理工学院（MIT）合作的计划；该计划要建立一套多使用者、多任务、多层次操作系统。为了完成满足这个要求的 UNIX 操作系统，1972 年，美国贝尔研究所的 D.M.Ritchie 推出 C 语言。这就是我们所熟知的 C 语言了，它有以下特点。

（1）既具有高级语言的特点，又具有汇编语言的特点。

（2）它可以作为工作系统设计语言，编写系统应用程序，也可以作为应用程序设计语言，编写不依赖计算机硬件的应用程序。

（3）它的应用范围广泛，具备很强的数据处理能力，不仅用于软件开发，而是各类科研都能用 C 语言。

3）面向对象程序设计时代

各种优秀的结构化程序设计语言相继出现，然而，随着软硬件环境逐渐复杂，20 世纪 60 年代爆发众所周知的软件危机，如何更好地提高软件开发效率并进行良好的维护渐渐成为当时程序设计领域面临的一个巨大挑战。面向对象程序设计就是在这样的背景下逐渐产生的。

面向对象程序设计的主要思想是将现实世界的物抽象成对象，现实世界中的关系抽象成类、继承，帮助人们实现对现实世界的抽象与数字建模。通过面向对象的方法，更利于用人能够理解的方式对复杂系统进行分析、设计与编程。

了解一下有哪些面向对象的语言。随着 C 语言的广泛普及和面向对象概念的出现，人们迫切希望在 C 语言中出现面向对象元素。1983 年，贝尔实验室的 Bjarne Stroustrup 推出了 C++。C++ 保留了 C 语言原有的所有优点，增加了面向对象的机制。因为 C++ 是由 C 语言发展而来的，所以 C++ 与 C 语言兼容，所以 C++ 既可用于面向过程的结构化程序设计，又可用于面向对象的程序设计。

C++ 的优点和缺点几乎是各占一半，如表 3-3 所示。

表 3-3　C++ 语言的优点与缺点对比

优　　点	缺　　点
高效，可移植	语义难以理解
给程序员更多的编程风格选择	正确性难以保证
开发成本优势明显	语言过于复杂
不会带来额外开销	

20 世纪 90 年代，SUN 公司预料未来科技将在家用电器领域大显身手，原本准备使用 C 语言，但 C 语言缺少垃圾回收系统，不具备可移植的安全性，没有分布程序设计和多线程功能。1995 年，SUN 推出了 Java 程序设计语言和 Java 平台（即 JavaSE、JavaEE、JavaME），获得了很大的成功。

我们了解 Java 语言的优缺点，如表 3-4 所示。

表 3-4　Java 语言优点与缺点对比

优　　点	缺　　点
纯面向对象	持续修改导致架构破坏
与平台无关	不能和操作系统底层打交道
解释性	代码相对冗长
多线程	
安全	
动态	
垃圾回收机制	

Java 编程语言的风格十分接近 C 语言和 C++ 语言，是一个纯粹的面向对象的程序设计语言，它继承了 C++ 语言面向对象技术的核心，舍弃了 C 语言中容易引起错误的指针（以引用取代）、运算符重载（operator overloading）、多重继承（以接口取代）等特性，增加了垃圾回收器功能用于回收不再被引用的对象所占据的内存空间，使得程序员不用再为内存管理而担忧。Java 通常被用在网络环境中，为此，Java 提供了一个安全机制以防恶意代码的攻击；Java 语言的设计目标之一是适应动态变化的环境，Java 程序需要的类能够动态地被载入到运行环境，也可以通过网络载入所需要的类；Java 不同于一般的编译执行计算机语言和解释执行计算机语言，它首先将源代码编译成二进制字节码（bytecode），然后依赖各种不同平台上的虚拟机来解释执行字节码；在 Java 语言中，线程是一种特殊的对象，它必须由 Thread 类或其子（孙）类来创建。通常有两种方法来创建线程：其一，使用型构为 Thread（Runnable）的构造子将一个实现了 Runnable 接口的对象包装成一个线程；其二，从 Thread 类派生出子类并重写 run 方法，使用该子类创建的对象即为线程。值得注意的是 Thread 类已经实现了 Runnable 接口，因此，任何一个线程均有它的 run 方法，而 run 方法中包含了线程所要运行的代码。线程的活动由一组方法来控制。Java 语言支持多个线程的同时执行，并提供多线程之间的同步机制。

1989年圣诞节期间，在阿姆斯特丹，Guido为了打发圣诞节的无趣，决心开发一个新的脚本解释程序。Python语言就这样诞生了，Python在设计上坚持了清晰划一的风格，这使得Python成为一门易读、易维护，并且被大量用户所欢迎的、用途广泛的语言。设计者开发时总的指导思想是，对于一个特定的问题，只要有一种最好的方法来解决就好了。Python语言常被昵称为胶水语言，它能够很轻松地把用其他语言制作的各种模块（尤其是C/C++）轻松地联结在一起。Python语言的这些指导思想同样是它的特点。它是完全面向对象的，而且可扩充，常见的一种应用情形是，使用Python快速生成程序的原型（有时甚至是程序的最终界面），然后对其中有特别要求的部分，用更合适的语言改写，例如，3D游戏中的图形渲染模块，速度要求非常高，就可以用C++重写。

Python语言的优点有很多，它简单易学、速度快，是一个免费开源的面向对象的高层语言，具有可移植性、解释性、可扩展性、可嵌入性、丰富的库，而且代码很规范。

3.2.3 类型语言

高级语言又可按照语言类型的强弱分为强类型语言和弱类型语言。

1. 强类型语言

强类型语言是一种总是强制类型定义的语言，要求变量的使用要严格符合定义，所有变量都必须先定义后使用。FORTRAN、Pascal、C、C++、Java、Python为强类型语言。

来看一个例子。

假如有一个整数，如果不显式地进行转换，你不能将其视为一个字符串。

```
int a=10;
string b=a+"abc";// 错，需要强制转换
```

2. 弱类型语言

弱类型语言是不总是要求变量强制类型定义的语言，BASIC、PHP为弱类型语言。

强弱类型语言优缺点比较：强类型语言写法相当麻烦，但因为有严格定义，所以不容易出错；弱类型语言代码简单，但因为变量没有确定的类型，所以容易出错。

3.2.4 程序设计语言的选择

说了很多种语言，到底什么时候用什么语言呢？该如何选择目前市面上常用的程序设计语言呢？在科学计算领域，现有程序用得较多的是FORTRAN语言，也有较少一部分用ALGOL语言；在企业级开发中，Windows平台上运用得最多的是.NET，跨平台用的是Java；在网站开发中PHP、Ruby多运用于后台的开发，JavaScript、FLEX、HTML5多运用于前台的开发；Python、Ruby、Lua多运用于嵌入式中；需要较高性能要求时，多运用C和C++语言；在移动平台开发上，Java运用在Android平台上，Objective C运用在iOS平台上。

3.3 C程序设计语言

C语言自诞生以来就一直引人关注，并很快形成全面、系统的标准。各个C语言编译器都遵循相同的标准，因此程序员可以在不同的编译器、不同操作系统中完成C语言

程序的开发。

C 语言是一种结构化的程序设计语言，它简明易懂，功能强大，适合于各种硬件平台，与常见的高级语言不一样的是，C 语言兼有高级语言和低级语言的功能。既可用于系统软件的开发，也适合于应用软件的开发。

3.3.1　C 语言特点

C 语言的特点表现在以下几个方面。

1. 程序设计结构化

结构化就是将程序的功能进行模块化，每一个模块具有不同的功能，程序将一些不同功能的模块有机地组合在一起，通过模块之间的相互协同工作，共同完成程序所要完成的任务。这种模块化的程序设计方式使得 C 语言程序易于调试和维护。

2. 运算符丰富

C 语言共有 34 种运算符。它把括号、赋值、逗号等都作为运算符处理，从而使 C 语言的运算类型极为丰富，可以实现其他高级语言难以实现的一些运算。

3. 数据类型全面

C 语言除了具有系统本身规定的一些数据类型外，还允许用户自定义数据类型，以满足程序设计的需要。

4. 书写灵活

只要符合 C 语言的语法规则，程序书写的格式并不受严格的限制。

注意：实际编写程序时并不提倡这样做，而是要求根据语法规则按缩进格式书写程序。

5. 适应性广

C 语言程序生成的目标代码质量高，程序执行效率高，与汇编语言相比，用 C 语言编写的程序可移植性好。

6. 关键字简洁

在 C 语言中，关键字有其特殊的意义和作用，不允许用户将其用作其他用途，所有关键字都必须是小写英文字母。

ANSIC 规定 C 语言共有 32 个关键字，如表 3-5 所示，其中：

（1）数据类型关键字 12 个。

（2）控制语句关键字 12 个。

（3）存储类型关键字 4 个。

（4）其他关键字 4 个。

表 3-5　32 个关键字

auto	break	case	char	const	continue	default	do
double	else	enum	extern	float	for	goto	if
int	long	register	return	short	signed	sizeof	static
struct	switch	typedef	union	unsigned	void	volatile	while

1999年12月16日，ISO推出了C99标准，该标准新增了5个C语言关键字，如表3-6所示。

表3-6　C99新增的关键字

inline	restrict	_Bool	_Complex	_Imaginary

2011年12月8日，ISO发布C语言的新标准C11，该标准新增了7个C语言关键字，如表3-7所示。

表3-7　C11新增关键字

_Alignas	_Alignof	_Atomic	_Static_assert	_Noreturn	_Thread_local	_Generic

C11标准中：

数据类型9个，其中char、int、float、double、void用于声明数据类型；short、long用于声明整型数据的大小；signed、unsigned用于声明整型数据在正负坐标上的区间。

自定义的数据类型3个，其中struct用于声明结构数据类型；union用于声明联合数据类型；enum用于声明枚举数据类型。

if、else、switch、case、default用于分支结构；for、while、do-while用于循环结构；continue用于结束本次循环，进入下一轮循环；break用于直接跳出循环结构或者分支结构；goto用于直接转移到指定的语句处；return用于返回到函数调用处。

auto用于声明自动变量；extern用于声明外部变量；register用于声明寄存器变量；static用于声明静态变量。

const用于声明只读变量；sizeof用于计算数据类型长度；typeof用于给自定义数据类型取别名等；volatile变量用于在程序执行中可被隐含地改变。

需要说明的是，除了上述的关键字以外，不同的实现环境对C语言的关键字有所扩充，并且扩充的关键字会因实现环境的不同而不同，读者只需要从使用的实现环境中去了解即可，在此不多加叙述，扩充的关键字只适合于特定的实现环境。

7. 控制结构灵活

C语言的程序结构简洁高效，使用方便、灵活，程序书写自由。C语言一共有9种控制结构，可以完成复杂的计算，9种控制结构及作用如表3-8所示。

表3-8　9种控制结构及作用

关键字	作　用	关键字	作　用	关键字	作　用
goto	直接转移	for	循环语句	break	直接跳出循环结构或分支结构
if	条件分支	do-while	循环语句	continue	结束本次循环，开始下一轮循环
switch	多路分支	while	循环语句	return	返回到函数调用处

了解C语言上述的特点，对学习和掌握好C语言程序设计很有帮助。虽然C语言程序对书写的要求没有太多的限制，只要符合语法规则就行，但在这里我们强调程序书写必

须规范，特别是初学者，这一点很重要。一个书写规范、整齐的 C 语言程序能够帮助程序员快速读懂程序所表达的思想，同时也能更清晰地将程序设计的意图正确地表达出来。

3.3.2　C 语言编程规范

在学习任何一种编程语言的时候，按照一定的规范培养良好的编程习惯很重要。良好的编程规范可以帮助开发人员理清思路、整理代码，同时也便于他人阅读理解代码从而促进交流。在进行 C 语言程序设计时，应该注意以下几方面的编程规范。

1. 空行

（1）在每个函数、结构体、枚举定义结束后应该加空行。

（2）在一个函数体内，逻辑密切相关的语句之间不加空行，其他地方应加空行分隔。

（3）相对独立的程序块之间、变量说明之后必须加空行。

2. 代码行

（1）一行代码只做一件事，如只定义一个变量，或只写一条语句，这样代码易于阅读及注释。

（2）if、for、while、do 等语句自占一行，执行语句不得紧跟其后。无论执行语句有多少都要加“{}”，这样可以防止书写错误。

3. 代码行内的空格

（1）关键字之后要留空格。

（2）函数名之后不要留空格，紧跟左括号“(”，以与关键字区别。

（3）“(”向后紧跟，“)”“,”“;”向前紧跟，紧跟处不留空格。

（4）“,”之后要留空格，如 Function(x,y,z)。如果“;”不是一行的结束符号，其后要留空格，如 for(i=0；i<10；i++)。

（5）赋值操作符、比较操作符、算术操作符、逻辑操作符的前后应当加一个空格。

（6）一元操作符，如“!”“~”“++”“--”“&”（地址运算符）等前后不加空格。

（7）像“[]”“.”“->”这类操作符，前后不加空格。

4. 对齐缩进

（1）程序块要采用缩进风格编写。

（2）对齐使用 Tab 键，Tab 键设置不同造成排版不同，应注意某些编辑器在识别、显示 Tab 键上存在问题，最终排版应以在项目的主代码编辑器中显示一致、整洁、清晰为准。

（3）函数或过程的开始，结构的定义及循环、判断等语句中的代码都要采用缩进风格，case 语句下的情况处理语句也要遵从语句缩进要求。

5. 标识符命名要求

（1）命名只能包含数字、字母、下画线。

（2）不能以数字开头。

（3）不能使用关键字作为名称。

（4）标识符的命名要清晰明了，有具体的含义，例如：使用完整的单词或单词缩写。

（5）函数命名，函数名中每个单词首字母大写；为避免由于函数名过长造成难以理解，可以在适当的位置使用下画线进行分割；命名中的前缀、关键缩写词等可以适当地采取全部大写。

（6）常量名全部大写。

（7）局部变量、全局变量、参数变量、成员变量，变量名一律小写，单词间使用下画线相连。

（8）静态全局变量使用 s_ 前缀，普通全局变量使用 g_ 前缀。

（9）宏命名全部大写，单词间使用下画线相连。

遵循 C 语言编程规范，可以养成良好的编码习惯，摒弃那些可能存在风险的编程行为，编写出安全健壮的代码，进而保证产品的可靠性、稳定性、安全性，增加软件的可读性，便于维护。遵循良好的编程规范，也是项目开发中相互协作的技术基础。

3.4 本章小结

本章主要讲解了程序的概念、流程图的绘制、程序设计语言的发展历程和 C 语言的主要编程规范。掌握流程图后，可以用流程图直观地展示程序的执行过程。了解程序设计语言的发展过程是了解编程语言的基础，掌握 C 语言编程规范可以让编程更加条理清晰、结构完整。

关键点概括如下。

（1）程序的概念。程序可以指一连贯的活动、作业、步骤、决断、计算和工序，当它们依照严格规定的顺序发生时即可实现特定目标或解决特定问题。

（2）用流程图来表示程序。用流程图来表示程序需要掌握流程图的符号及含义，将流程图的符号按顺序连接成表示程序的流程图。

（3）程序设计语言的发展阶段分为前计算机时代、机器语言时代、汇编语言时代以及高级语言时代。

（4）类型语言分类：强类型语言和弱类型语言。

（5）C 语言是结构化语言，功能齐全，适用范围广泛，可以对硬件进行操作，文件操作能力强。C 语言强大的功能和友好的编程风格使它成为最流行的编程方式，并且经久不衰。

（6）C 语言是一种结构化的程序设计语言，在书写代码的时候应该遵循编程规范。这样有助于查看程序的条理，帮助设计者厘清思路，也便于他人阅读。

3.5 本章习题

1．简单描述“大一新生报名时办理入学手续”的流程，并绘制出对应流程图。

2．场景描述：张伯伯计划本周末带自己的宠物猫去宠物医院洗澡，但是宠物医院每天只接受 30 只宠物洗澡预约。张伯伯首先通过手机预约周六宠物洗澡服务，但由于预约已满，张伯伯不得不选择预约周日，还好周日没有排满，张伯伯最终顺利预约到第 18 名。要求按照上述场景描述梳理出整个流程，并绘制出对应流程图。

3．现有 8 个固定纸盒，每个纸盒中放入任意一个正整数，不允许挪动已有纸盒位置(纸盒中的数字可以任意拿放,但一个纸盒最多只能放一个数字),可以增加新的纸盒，要求将这 8 个正整数从小到大依次排序，请给出解决这个问题的算法。

4．任意输入一个数 n（正整数），用 sum 表示 1 到 n 的累加和，并输出 sum，请给出解决这个问题的算法并绘制出相应流程图。

5．简述程序设计语言的发展过程和主要阶段。

6．简述对 C 程序设计语言的认识。