信息技术

教师参考书

八年级　下册

王超智　主编

主　　编：王超智
分册主编：常庆卫 段志起
编写人员：张旭臣 徐青青 朱安海 鲁　晶
　　裴志刚 马惠超 应合林 马云众
　　刘军芳 马开生 桂　元

责任编辑：李宏富 赵丽洁

编 写 说 明

为了配合河南大学出版社、河南电子音像出版社出版的2020年版《信息技术》教材的使用，帮助教师理解教材的编写思想并领会作者的编写意图，更好地把握教材，我们组织教材的作者编写了这套《信息技术教师参考书》。

《信息技术教师参考书》首先在各册内容前针对相应册次的教材做了总体说明，旨在帮助教师在最短时间内了解相应册次教材的结构、内容及编写思路；然后根据教材的章节顺序对教材内容进行详细分析，提出教学建议，给出实践活动指导、思考与探索及练习提升的参考答案。

《信息技术教师参考书》对每一节教材内容都做了详细的分析并提出了教学建议，具体内容如下。

（1）教材分析。这是针对一节教材内容所做的具体分析，旨在帮助教师把握教材内容。其下又分为内容分析、教学目标、教学重点和难点三个方面：内容分析是针对教材内容所做的具体分析，包括设置活动的意图等，可帮助教师准确把握教材；教学目标是本节教学需要达成的目标，可帮助教师明确教学的目标；教学重点和难点是本节教学的重点内容和难点内容，可帮助教师明确把握重点，并提醒教师要想方设法突破难点。

（2）学情分析。这是针对学生情况进行的分析，旨在帮助教师了解学生已掌握的知识和能力水平，以及未掌握的、想要知道的知识。通过这些信息，教师能及时调整教学方案，提高教学效率，确保教学活动不偏离预定的教学目标。

（3）教学建议。这是教材作者针对一节教材的教学所提的建议，仅供教师上课时参考。教师不要拘泥于本教参提出的教学建议，可根据学生的具体情况和学校的设施情况大胆尝试其他教学方法。其下又分为课前准备、过程设计。课前准备是为了提高教学效率、达成教学目标，教师和学生需要做的各项准备工作，包括设备、学生和教师的准备。过程设计是针对本节教学过程的建议，仅供参考。

（4）实践活动指导。这是针对教材中的“实践活动”所做的指导，旨在帮助提升教师的专业能力，便于对学生进行有效指导。

（5）思考与探索参考答案。这是针对教材中的“思考与探索”所给出的参考答案、

参考思路、参考方法等，都以“解析”的表述形式提供。

（6）练习提升参考答案。这是针对教材课后的“练习提升”给出的参考答案、参考思路、参考方法等，都以“解析”的表述形式提供。

（7）教师知识延伸。这是根据教学内容提供的教师需要拓展的相关知识，供教师扩大知识面。由于信息技术发展很快，部分知识、概念、理论和方法可能会随着时间而变化，教师要注意收集最新的相关知识和信息。

信息技术是发展中的学科，新的教学理念、新的知识和新的技术需要在教学实践中不断检验和完善。我们诚恳地希望各位教师在使用教材和教师参考书的过程中，对其中存在的缺点和错误及时批评、指正，以利于我们修正。

在本册《信息技术教师参考书》中，引用教材中的图片或表格序号时，我们使用“-”作为连接，以与教材相统一，如：“图 1-1-1”表示教材第一章第一节第 1 个图，“表 1-1-1”表示教材第一章第一节第 1 个表格。《信息技术教师参考书》中的图片或表格序号用“.”作为连接，如：“图 1.1.1”表示《信息技术教师参考书》第一章第一节第 1 个图，“表 1.1.1”表示《信息技术教师参考书》第一章第一节第 1 个表格。

编　　者

本 册 综 述

《信息技术 八年级 下册》内容是程序设计，对应《中小学综合实践活动课程指导纲要 7－9 年级》的活动主题“走进程序世界”，主要教学目的是让学生掌握程序设计的基本方法，理解算法思想，并能掌握利用编程解决问题的常用策略。

“计算思维是运用计算机科学的基础概念进行问题求解、系统设计、以及人类行为理解等涵盖计算机科学之广度的一系列思维活动。”教材在计算思维培养方面，主要体现在提高学生问题求解能力，增强学生编程思维活动。

同许多“某某语言程序设计”的教材不同，本册教材不是介绍或学习某种计算机语言，也不是介绍计算机语言编译或解释环境（IDE）的使用方法，更不是介绍某种计算机语言的特点和优势。本册教材的教学目标并不是让学生掌握某种计算机语言编写程序，原因是计算机语言是随着时代的发展不断更新的，不论现在学习哪一门计算机语言，都不能保证在以后的学习或工作中能够继续使用。现在学习的计算机语言可能在不久之后就会过时，甚至被淘汰。例如，笔者以前在教学中讲授的 PASCAL、BASIC 语言或者 VB 开发工具等，并不是这些语言或开发工具不好，而是随着计算机技术的发展及市场需求的变化，人们对计算机语言的需求在不断变化。掌握这些相对过时的计算机语言或开发工具对学生建立计算思维的意义很有限，甚至会有负面作用。因此，本册教材在编写过程中尽量避免成为某某计算机语言教材或某某开发工具的使用说明书。

但是，进行程序设计的学习必须通过某种计算机语言来实现，本册教材提供了 JavaScript 和 Python 两种计算机语言的程序范例，以供学习者选择使用。大家学习的时候，不要过于关注这两种计算机语言本身的特性，而要更注重于程序设计思想的学习。原则上，这两种计算机语言大家只需要选择其中一种就可以完成本书所有程序设计内容的学习，如果教师认为这两种计算机语言都不好，也可以考虑选择其他计算机语言来进行教学，只需将本书中的程序范例用该计算机语言替代即可。本书所教授的程序设计内容，尽量避开计算机语言的个体特性，所讲授的内容是大多计算机语言所共有的。因此，采用其他计算机语言进行教学时，也能在其中找到相应内容。

目　　录

第一章　认 识 程 序

本章的主要内容是介绍计算机语言的基本概念，通过实践活动完成程序设计的一些基本操作，让学生了解创建计算机程序并运行程序的基本方法步骤。通过输入、输出等几条基本语句的书写，让学生有通过计算机程序控制计算机行事的基本意识。

教学课时建议：3~4 课时。

本章共三节，分别达到：掌握录入、运行程序的方法，学会编写输出语句、赋值语句，学会编写输入语句的教学基本目标。

第一节　与计算机交流
——认识计算机语言

一、教材分析

（一）内容分析

本节主要通过操作计算机程序代码的录入、保存、运行的完整过程，让学生主观感受计算机程序的作用，消除对计算机程序设计的陌生感，增强对程序设计的学习信心。

（二）教学目标

（1）通过编辑、保存、运行程序的课堂活动，了解计算机程序设计的意义。

（2）通过编辑、保存、运行程序的课堂活动，理解计算机执行程序指令的过程。

（3）通过编辑、保存、运行程序的课堂活动，掌握运行计算机源代码程序的方法。

（三）教学重点和难点

1. 教学重点

（1）计算机源代码程序的录入和保存的方法。

（2）计算机源代码程序运行及运行过程中人机交互的方法和步骤。

2. 教学难点

计算机源代码程序中错误的查找和调试。

二、学情分析

学生在学习本节之前，对计算机的基本操作已经有了一定了解，也用图形化程序设计工具设计过程序，但对程序代码仅仅做到了解。学生习惯于对已经编译打包过的应用程序进行操作，而不会对代码源程序进行任何操作。初次接触完全陌生的程序设计内容难免会有畏难情绪和一定的抵触心理，我们要通过源程序操作的简便性让学生了解计算机程序执行的原理和过程，从而克服对程序设计的神秘感，树立学会程序设计的信心，从而顺利进入学习状态。

三、教学建议

本节教学建议 1－2 学时。

对于“阅读拓展”模块中“计算机语言简介”部分的内容，可根据学生接受情况有选择性地教学，如果学生对计算机比较陌生，可以略过；如果学生接受情况良好，可以增加 1 课时用于“计算机语言简介”部分的讲解，并可加入代码错误检查，即程序调试的相关内容，让学生进一步熟悉程序运行环境。

说明：对于“知识窗”部分，在以后教学中使用 Python 的可以进行学习，否则可以略过。这部分需要软件环境的支持，需要教师提前在学生机上安装 Python 3.x 的软件工具，这一工具可以在 Python 官网上下载，需要教师提前熟悉这个软件开发环境。可以使用 Python IDLE（IDLE is Python's Integrated Development and Learning Environment）集成开发学习环境来运行和调试程序。

（一）课前准备

设备：计算机教室，建议使用 JavaScript 教学，软件环境是 Windows 操作系统常用的“记事本”“IE 浏览器”或教师熟悉的其他浏览器；如果使用 Python，须提前在学生机上安装 Python 3.x IDE 软件。

学生：预习本节内容。

教师：在教师机上安装所选软件，熟悉软件使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

根据不同个性化的需求（例如：用计算机统计现场评分），有现成的软件可以使用吗？如果没有，可以通过程序设计来实现，从而引出程序设计的概念。

2. 实践活动，掌握技巧

通过录入程序代码、保存、运行等一系列实践活动的具体操作，让学生体验程序设计解决问题的过程。具体代码不要求学生完全理解，但是，必须让学生明白解决问题的关键是程序代码的编写。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结，时间允许可以针对“阅读扩展”内容进行扩展讲解。

四、思考与探索参考答案

浏览器究竟是什么？通常网络浏览器是用来浏览网页的，现在为什么可以运行程序了呢？

解析：

浏览器本身是一个用来浏览网络的软件，它还有一个功能是解释 JavaScript、VBScript 脚本程序及 HTML 代码。在本节中，用到的就是浏览器的这个功能，浏览器起到了计算机脚本程序解释器的作用。

五、练习提升参考答案

将程序中的 n=12 改成 n=5，再运行程序，会出现什么变化？

解析：

将程序中的 n=12 改成 n=5，再运行程序，只能对 5 位评委的打分进行统计。

六、教师知识延伸

JavaScript 程序调试方法

假设教材中程序以“1-1a.htm”为文件名存储在 C 盘根目录下。下面以 IE11 为例，进行介绍。

在地址栏中输入：“C:\1-1a.htm”后，敲回车键运行程序。

1. 可能出现的错误情况及解决的方法

根据 IE11 设置的不同，可能出现不同的情况。

如出现如图 1.1.1 所示的提示，单击“允许阻止的内容”。

图 1.1.1　出现“允许阻止的内容”提示

如果程序正确，会按教材中的提示运行程序；如果程序不正确，根据程序错误的不同，会有不同的显示。

调试中可能会多次运行程序，用 F5 刷新浏览器较为方便。下面，就几种常见的错误程序运行情况进行说明。

（1）错误情况 1：浏览器没有执行程序，而是直接显示部分程序代码，如图 1.1.2 所示。

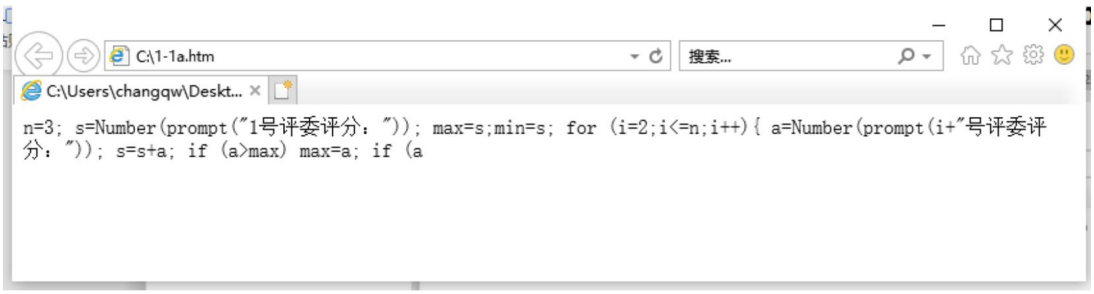

图 1.1.2　错误情况 1

错误原因及解决方法：

<SCRIiPT>

. . . . . .

</SCRIPT>

如上程序头出现错误，修改成正确的即可。

（2）错误情况 2：浏览器执行程序没有任何显示。

这种情况可能是程序出现语法错误。语法错误是指程序代码不符合程序设计语言的格式要求，造成不能被编译或者解释。出现这类错误，需要认真检查程序代码，纠正错误。如果程序代码较多，而错误又是肉眼难以发现的，例如程序中的半角分号“;”误写成了全角分号“；”，两者看着相似，对程序来说却是质的区别。半角的分号是 JavaScript 的语句分隔符，而全角的分号是中文中的一个符号，计算机是不会将其识别为语句分隔符的，只会出现运行错误。找到此类错误确实有难度，更好的方法是通过调试窗口来查错。

（3）错误情况 3：浏览器执行程序有显示，但不是正确的运行结果。

这种情况可能是程序代码有语义错误或逻辑错误，即程序代码符合程序设计语言的要求，能够被编译或解释，也能够被执行，但程序执行的结果不符合要求。出现这类错误，需要认真检查程序代码及执行流程，思考结果不符合要求的原因和在程序中的位置，然后纠正错误。同样，如果程序代码较多，发现错误很不容易。因此，浏览器提供了调试窗口

来帮助查错。

2. 用调试窗口调试程序的方法

调试程序的目的是为了发现程序中存在的各种问题。发现问题后要修正程序，以使程序对所有的输入都能正确运行，并输出正确的结果。

用调试窗口查错的方法如下。

第 1 步：按 F12 键，弹出 JavaScript 的调试窗口。

第 2 步：设置断点。单击需要设置断点的语句前面的语句标号，设置断点。

第 3 步：单击“启动调试”，程序执行到设置有断点的语句，进入等待状态。

第 4 步：按 F10 键单步执行程序，在监视窗口中通过输出语句可显示每条语句执行后变量的值。按 F8 键，继续执行程序。如果程序再次执行到断点语句行，就会重新中断程序的执行，进入等待状态。

当断点完成后就会回到 IE 窗口，这时变量保持程序执行后的值。如图 1.1.3 所示。

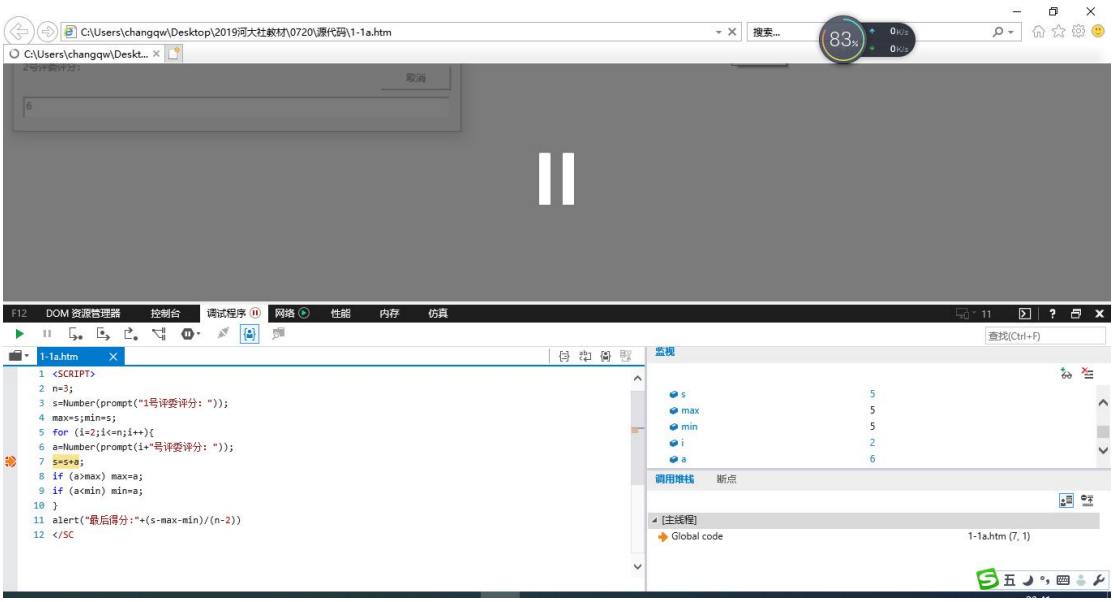

图 1.1.3　调试程序窗口

第二节　初识程序的奥秘（上）
——保留字和变量

一、教材分析

（一）内容分析

本节通过对计算机语言的基本知识介绍，让学生从细节上进一步熟悉计算机程序的构成和实现意义。

本节将通过让学生录入几个简单的程序，并尝试进行一些改动和调整来进一步了解程序，巩固上节的内容，达到熟练掌握程序的录入、保存和执行的方法和步骤。

（二）教学目标

（1）通过运行几个带有输出语句、赋值语句的程序，掌握输出语句、赋值语句的使用方法。

（2）通过运行几个带有输出语句、赋值语句的程序，并比较其中的语句，理解变量、语句在程序中的作用。

（3）通过讲解，了解计算机中的保留字、表达式。

（三）教学重点和难点

1. 教学重点

（1）输出语句、赋值语句在计算机中的执行过程和使用方法。

（2）变量的概念及变量在计算机中的存储和读写过程。

2. 教学难点

（1）变量的概念和赋值语句的执行过程。

（2）计算机源代码程序中错误的查找和调试。

二、学情分析

本节还是学生接触计算机代码程序的开始阶段，主要任务仍是通过强化源程序操作，

让学生进一步熟悉操作过程和步骤，更快进入学习状态。本节的教学内容并不是太难，一半的教学任务仍是让学生进一步熟悉程序的书写、保存和执行步骤，以及软件环境。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

分析上节录入并运行的程序代码，根据学生的现有程序水平和知识，不会完全知道程序代码的具体意义，从而引出程序语言基本知识的相关概念，如变量、语句等。

2. 实践活动，掌握技巧

通过“实践活动”，录入几个只有一条输出语句的程序代码，并用上节所学的方法保存、运行程序，让学生掌握输出语句的使用方法。在活动过程中，会出现一些问题，通过解决这些问题，让学生掌握计算机语言中表达式的写法，并初步理解数据类型的概念。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结，若时间允许可以针对“阅读扩展”内容进行扩展讲解。

四、思考与探索参考答案

想一想：若变量 a 的值为 5，JavaScript 的输出语句 alert("a")与 alert(a)或 Python 的输出语句 print("a")与 print(a)，哪一个能输出 5?

解析：

alert(a)与 print(a)能输出 5。如果加了双引号，只能输出字母 a，不能输出变量 a 的值。

五、练习提升参考答案

第 1 题：变量的本质是_____。

A．一个英文字母或单词　　B．一个数值

C．一个字符串　　D．内存的存储单元

解析：

D

第 2 题：下面程序的运行结果是_____。

JavaScript 程序如下:

```
<SCRIPT>
a=3;
b=5;
a=b;
alert(a);
</SCRIPT>
```

Python 程序如下：

```
a=3;
b=5;
a=b;
print(a);
```

A．3　　B．5　　C．8　　D．0

解析：

B

第 3 题：数学表达式 5÷[2×(3+6)+(7-3)]在高级语言中应表示为_____。

A．5/[2*(3+6)+(7-3)]　　B．5/(2*(3+6)+(7-3))

C．5/[2×(3+6)+(7-3))　　D．5/[2*(3+6)+(7-3)]

解析：

A．不能用中括号[]

B．正确

C．不能用数学中的乘号

D．不能用中括号[]

答案：B

六、教师知识延伸

JavaScript 技术参考（一）

（一）JavaScript 的显示方案

JavaScript 能够以不同方式“显示”数据：

使用 window.alert()写入警告框；

使用 document.write()写入 HTML 输出；

使用 console.log()写入浏览器控制台。

（二）JavaScript 的变量

JavaScript 变量是存储数据值的容器。

【例 1】x、y 和 z 是变量。

```
var x=7;
var y=8;
var z=x+y;
```

从上例中，可得：

x 存储值为 7，y 存储值为 8，z 存储值为 15。

【例 2】price1、price2 和 total 是变量。

```
var price1=7;
var price2=8;
var price3=12;
var total=price1+price2+price3;
```

在编程中，类似代数，我们使用变量（如 price1）来存放值。

在编程中，类似代数，我们在表达式中使用变量，如：total=price1+price2。

从上例中，可以算出 total 的值是 15。

（三）JavaScript 的标识符

所有 JavaScript 的变量必须以唯一的名称来标识。这些唯一的名称称为标识符。

标识符可以是短名称（如 x 和 y），也可以是更具描述性的长名称（如 age、sum、totalVolume）。

构造变量名称（唯一标识符）的通用规则是：

名称可包含字母、数字、下划线和美元符号；

名称必须以字母开头；

名称也可以$和_开头（但是在教材讲解中我们不这么做）；

名称对大小写敏感（y 和 Y 是不同的变量）；

保留字（比如 JavaScript 的关键词）不能用作变量名称。

提示：JavaScript 的标识符对大小写敏感。

（四）声明（创建）JavaScript 变量

在 JavaScript 中创建变量称为“声明”变量。可以通过 var 关键词来声明 JavaScript 变量，如：

```
var carName;
```

声明之后，变量是没有值的。（技术上，它的值是 undefined）

如需赋值给变量，可使用赋值号（等号），如：

```
carName="比亚迪";
```

也可以在声明变量时对它赋值，如：

```
var carName="比亚迪";
```

在上面的例子中，我们创建了名为 carName 的变量，并向其赋值"比亚迪"。

提示：在脚本的开头声明所有变量是个好习惯！

一个语句，可以声明多个变量，以 var 作为语句的开头，以逗号作为变量间的分隔符，如：

```
var person="张三",carName="比亚迪",price=98000;
```

声明还可横跨多行，如：

```
var person="李四",
carName="比亚迪",
price=98000;
```

在计算机程序中，被声明的变量经常是不带值的。值可以是需被计算的内容，或是之后被提供的数据，比如数据输入。

不带有值的变量，它的值将是 undefined。

变量 carName 在语句 var carName;执行后的值是 undefined。

如果再次声明某个 JavaScript 变量，其原来的值仍会保留。在下面两个语句执行后，变量 carName 的值仍然是"比亚迪"。

```
var carName="比亚迪";
var carName;
```

（五）JavaScript 的算术运算

与代数类似，我们能够通过 JavaScript 变量进行算术运算，可使用=和+之类的运算符：

```
var x=3+5+8;
```

字符串也可以使用加号，但是字符串将被级联，如：

```
var x="张"+" "+"三";
```

字符串和数值也可以使用+号连接，这里的数值将被视作字符串，进行级联，如：

```
var x="8"+3+5;
```

提示：如果把数值放入引号中，其余数值会被视作字符串并被级联。

（六）JavaScript 注释

JavaScript 注释用于解释 JavaScript 代码，增强其可读性。

JavaScript 注释也可以用于在测试替代代码时阻止其执行。

1. 单行注释

单行注释以//开头，任何位于//与行末之间的文本都会被 JavaScript 忽略（不会执行）。

可在每代码行之前使用单行注释，以说明下面代码的作用、采用的算法等，如：

```
//显示标题：
document.write("我的第一个程序");
//改变段落：
document.write("我的<br>第一个<br>程序");
```

还可以在每行代码之后使用单行注释，以对本行代码进行注释，如：

```
var x=5;            //声明 x，为其赋值 5
var y =x+2;       //声明 y，为其赋值 x+2
```

2. 多行注释

多行注释以/*开头，以*/结尾，任何位于/*和*/之间的文本都会被 JavaScript 忽略。多行注释（注释块）如下：

```
/*
  下面的代码中的<br>会改变段落：
*/
document.write("我的第一个程序");
document.write("我的<br>第一个<br>程序");
```

注释：使用单行注释最为常见。

3. 使用注释来阻止执行

使用注释来阻止代码执行很适合代码测试。在代码行之前添加//会把可执行的代码行更改为注释，以阻止本行代码的执行，如：

```
//document.write("我的第一个程序");
document.write("我的<br>第一个<br>程序");
```

还可以使用注释块来阻止多行代码的执行，如：

```
/*
document.write("我的第一个程序");
document.write("我的<br>第一个<br>程序");
*/
```

Python 技术参考（一）

（一）直接执行

可以直接在命令行中输入 Python 的语句，回车后，执行结果直接显示，如：

```
>>>print("Hello,World!")
Hello,World!
```

还可以在服务器上创建 Python 程序文件（扩展名为 py），然后在命令行中运行它：

```
C:\Users\Your Name>python myfile.py
```

（二）缩进

缩进是指代码行开头的空格。

在很多编程语言中，代码缩进仅仅为了提高程序的可读性，而 Python 中的缩进涉及程序的结构，影响程序的执行，显得非常重要。

Python 使用缩进来表示代码块，如：

```
if 5>2:
   print("Five is greater than two!");
```

如果省略缩进，Python 会报告出错，显示：

```
语法错误：
if 5 > 2:
print("Five is greater than two!");
```

空格的多少取决于程序员，但至少需要一个。如：

```
例 3：if 5>2:
          print("Five is greater than two!");
例 4：if 5>2:
               print("Five is greater than two!");
```

两个例子的含义和执行结果完全一样。

在同一代码块中必须使用相同数量的空格，否则 Python 会报告出错，如：

```
if 5>2:
  print("Five is greater than two!");
          print("Five is greater than two!");
```

（三）Python 变量

变量是存放数据值的容器。

1. 声明变量

与其他编程语言不同，Python 没有声明变量的语句，首次为其赋值时，会自动创建变量，如：

```
x=10;
y="张三";
print(x);
print(y);
```

变量不需要使用任何特定类型声明，甚至可以在设置后更改其类型，如：

```
x=5 # x is of type int;
x="Steve" # x is now of type str;
print(x);
```

字符串变量可以使用单引号或双引号进行声明，如：

```
x="张三";# is the same as
x='张三';
```

在 Python 中，变量是在为其赋值时创建的，如：

```
x=5;
y="Hello,World!";
```

2. 变量名称

变量可以使用短名称（如 x 和 y）或更具描述性的长名称（如 age、carname、total_volume）。

Python 变量命名规则：

（1）变量名必须以字母或下划线字符开头；

（2）变量名称不能以数字开头；

（3）变量名只能包含字母、数字、字符和下划线（A-z、0-9 和_）；

（4）变量名称区分大小写（age、Age 和 AGE 是三个不同的变量）。

切记：变量名称区分大小写。

3. 向多个变量赋值

Python 允许在一行中为多个变量赋值，如：

```
x,y,z="橘子","香蕉","樱桃";
print(x);
print(y);
print(z);
```

Python 还允许在一行中为多个变量分配相同的值，如：

```
x=y=z="橘子";
print(x);
print(y);
print(z);
```

4. 输出变量

Python 的 print 语句通常用于输出变量。如需结合文本和变量，Python 使用+字符，如：

```
x="awesome";
print("Python is "+x);
```

还可以使用+字符将变量与另一个变量相加，如：

```
x="Python is ";
y="awesome";
z=x+y;
print(z);
```

对于数字，+字符用作算术运算符，如：

```
x=5;
y=10;
print(x+y);
```

Python 不允许将字符串和数字用+连接，连接时 Python 会报告错误，如：

```
x=10;
y="Bill";
print(x+y);
```

5. 全局变量

在函数外部创建的变量（如上述所有实例）称为**全局变量**。

全局变量可以被函数内部和外部的每个函数使用。

在函数外部创建的变量，可在函数内部使用，如：

```
x="awesome";
def myfunc():
  print("Python is "+x);
myfunc();
```

如果在函数内部创建具有相同名称的变量，则该变量将是局部变量，并且只能在函数内部使用。具有相同名称的全局变量将保留原样，并拥有原始值，如：

```
x="awesome";
def myfunc():
  x="fantastic";
  print("Python is "+x);
myfunc();
print("Python is "+x);
```

（四）注释

可以对 Python 的语句、语句组或程序进行注释。

注释以#开头，Python 将其余部分作为注释呈现，如：

```
#This is a comment.
```

```
print("Hello,World!");
```

注释可用于说明 Python 代码，提高代码的可读性。在测试代码时，可以使用注释来阻止代码的执行。

1. 创建注释

注释以#开头，Python 将忽略它们，可放在一行的开头，如：

```
#This is a comment
print("Hello,World!");
```

注释也可以放在一行的末尾，如：

```
print("Hello, World!"); #This is a comment
```

注释还可以用来阻止 Python 执行代码，如：

```
#print("Hello,World!")
print("Cheers,Mate!");
```

2. 多行注释

Python 实际上没有多行注释的语法。

要添加多行注释，可以为每行插入一个#，如：

```
#This is a comment
#written in
#more than just one line
print("Hello, World!");
```

也可以利用 Python 的特点，以不完全符合预期的方式，使用多行字符串进行注释。由于 Python 将忽略未分配给变量的字符串文字，因此可以在代码中添加多行字符串（三引号），并在其中添加注释，如：

```
"""
This is a comment
written in
more than just one line
"""
print("Hello,World!");
```

只要字符串未分配给变量，Python 就会读取代码，然后忽略它，这样就相当于对多行进行了注释。

第三节　初识程序的奥秘（下）
——输入语句

一、教材分析

（一）内容分析

本节通过对计算机语言的输入语句介绍，让学生从人机交互的角度进一步熟悉计算机程序的构成和实现意义。

本节通过让学生录入几个简单的程序，并尝试进行一些改动和调整来进一步了解程序，并巩固前面的内容，加强记忆程序的录入、保存和执行的方法步骤。

（二）教学目标

（1）通过编写、运行冬奥会程序的课堂活动，掌握输入语句的使用方法。

（2）通过编写、运行冬奥会程序的课堂活动，掌握输入、输出语句在交互式程序中配合使用的技巧和方法。

（3）通过编写、运行冬奥会程序的课堂活动，理解计算机中数据类型的概念。

（三）教学重点和难点

1. 教学重点

（1）输入语句的计算机工作原理和使用方法。

（2）计算机数据类型的概念。

2. 教学难点

（1）数据类型的概念。

（2）计算机源代码程序中错误的查找和调试。

二、学情分析

学生在学习本节内容之前，对程序的输出操作已有一定了解，但对人机交互的实现是不完整的。学习了输入语句，学生会对程序的人机交互有一个相对完整的认识。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过回顾上节的知识要点，用“头脑”“嘴巴”“耳朵”来形容语句，从而引出“耳朵”的对应语句——输入语句。

2. 实践活动，掌握技巧

通过“实践活动”，录入几个不仅有输出语句，而且有输入语句的程序代码，并运行程序，让学生体验人与程序（机）进行交互的方法。在“思考与探索”中，会出现计算结果不符合要求的问题，这个计算错误是由于程序设计而产生的，通过解决这个问题，让学生理解计算机语言中数值和字符串这两种数据类型。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、思考与探索参考答案

执行如下程序。

JavaScript 程序：

```
<SCRIPT>
a=prompt("请输入加数：");
b=prompt("请输入加数：");
c=a+b;
alert(a+"+"+b+"="+c);
</SCRIPT>
```

Python 程序：

```
a=input("请输入加数：");
b=input("请输入加数：");
c=a+b;
print(a,"+",b,"=",c);
```

按屏幕提示键入两个加数（如键入 3 和 4）。

屏幕显示：_____

为什么？_____

_____

_____

如果要让计算机做乘法运算，程序应修改为：

_____

_____

_____

_____

解析：

按屏幕提示键入两个加数（如键入 3 和 4）

屏幕显示：34

为什么？因为计算机认为 3 和 4 是字符串，即"3"、"4"字符串加法的结果是将两个字符串相连接，结果为“34”。

如果要让计算机做加法运算，程序可如下修改。

JavaScript 程序：

```
<SCRIPT>
a=Number(prompt("请输入加数："));
b=Number(prompt("请输入加数："));
c=a+b;
alert(a+"+"+b+"="+c);
</SCRIPT>
```

Python 程序：

```
a=eval(input("请输入加数："));
b=eval(input("请输入加数："));
c=a+b;
```

print(a,"+",b,"=",c);

其实还有一种改写程序的方法：只将 c=a+b 一行改为 c=0+a+b，其他行不变。尽管这种改写更简单，但不建议在课堂上提及，让学生的学习注意力集中在 Number()和 eval()的使用上，如果有学生自己探索出来，再单独进行类型自动转换的说明。

五、练习提升参考答案

第 1 题：下列语句中不具备运算功能的是_____。

A．输出语句　　B．赋值语句　　C．输入语句

解析：

答案：C

提示：输出语句在输出表达式时是先计算后输出，赋值语句对右边表达式的处理也一样，先计算后赋值。

第 2 题：输入语句的主要功能是_____。

A．从输入设备（如键盘）输入数据，并存入存储器存储单元（变量）中

B．将输入数据进行计算，并存入存储器存储单元（变量）中

C．输入数据，经过计算后通过输出设备输出

D．从输入设备中读入计算指令，并在计算机中运行

解析：

答案：A

提示：输入语句的功能是读入数据，赋值语句的功能是计算。

六、教师知识延伸

JavaScript 技术参考（二）

（一）JavaScript 的数据类型

JavaScript 中的变量可存放数值，比如 100；还可以存放文本，比如"张三"。

在编程中，文本值被称为字符串。

JavaScript 可处理多种数据类型，但是现在，我们只关注数值型数据和字符串型数据。

放在双引号或单引号中的一串字符称作字符串，数值不用引号。

如果把数字放在引号中，会被视作字符串。如：

```
var pi="3.14";
var person="张三";
var answer='你好！';
```

（二）JavaScript 的动态数据类型

JavaScript 拥有动态数据类型，这意味着相同的变量可用作不同的数据类型。如：

```
var x;                  //x 为 undefined
var x=5;                //现在 x 为数值
var x="张三";       //现在 x 为字符串
```

（三）JavaScript 的字符串变量

字符串变量是存储字符串（比如"比亚迪"）的变量。

字符串可以是引号中的任意文本，可以使用单引号或双引号为字符串变量赋值。如：

```
var carname="长城";
var carname='长城';
```

在字符串中可以使用引号，但这里的引号不能与表示字符串的引号相匹配，如：

```
var answer="It's alright";
var answer="He is called 'Johnny'";
var answer='He is called "Johnny"';
```

（四）JavaScript 的数值

JavaScript 只有一种数值类型。数字可以带小数点，也可以不带。如：

```
var x1=34.00;     //使用小数点来写
var x2=34;        //不使用小数点来写
```

极大或极小的数字可以通过科学计数法（指数）来书写。如：

```
var y=123e5;      //12300000
var z=123e-5;    //0.00123
```

（五）JavaScript 的布尔类型

布尔（逻辑）类型只有 true（真）或 false（假）两个值。

```
var x=true;
var y=false;
```

布尔类型常用在条件测试中。

（六）变量的 Undefined 和 Null 值

变量的 Undefined 值表示变量不含有值，通过将变量的值设置为 null 可以清空变量。

```
cars=null;
```

```
person=null;
```

（七）声明变量类型

可以使用保留字 new 来声明变量的类型，如：

```
var carname=new String;
var x=new Number;
var y=new Boolean;
var cars=new Array;
var person=new Object;
```

JavaScript 的变量均为对象。当声明一个变量时，就创建了一个新的对象。

提示：JavaScript 具有隐含的全局概念，意味着不声明的任何变量都会成为一个全局对象属性。

Python 技术参考（二）

（一）Python 的数据类型

1. 内置数据类型

在编程中，数据类型是一个重要的概念。

变量可以存储不同类型的数据，并且不同类型变量或数据可以执行不同的操作。

在这些类型中，Python 默认拥有表 1.3.1 所示的内置数据类型。

表 1.3.1　Python 的内置数据类型

文本类型	str
数值类型	Int、floa
序列类型	List、tuple、range
映射类型	dict
集合类型	Set、frozenset
布尔类型	bool
二进制类型	Bytes、bytearray、memoryview

2. 获取数据类型

使用 type()函数可以获取任何对象的数据类型，打印变量 x 的数据类型：

```
x=10;
```

```
print(type(x));
```

3. 设置数据类型

在 Python 中，为变量赋值时，会设置数据类型，如表 1.3.2 所示。

表 1.3.2　Python 中为变量赋值时可设置数据类型

示　　例	数据类型
x="Hello World"	str
x=29	int
x=29.5	float
x=1j	complex
x=["apple", "banana", "cherry"]	list
x=("apple", "banana", "cherry")	tuple
x= range(6)	range
x={"name":"Bill","age":63}	dict
x={"apple","banana","cherry"}	set
x=frozenset({"apple","banana","cherry"})	frozenset
x=True	bool
x=b"Hello"	bytes
x=bytearray(5)	bytearray
x=memoryview(bytes(5))	memoryview

4. 设定特定的数据类型

如果希望指定数据类型，则可以使用表 1.3.3 所示的方法构造函数。

表 1.3.3　用构造函数为变量指定数据类型

示　　例	数据类型
x=str("Hello World")	str
x= int(29)	int
x=float(29.5)	float

续表

示　例	数据类型
x=complex(1j)	complex
x=list(("apple","banana","cherry"))	list
x=tuple(("apple","banana","cherry"))	tuple
x=range(6)	range
x=dict(name="Bill",age=36)	dict
x=set(("apple","banana","cherry"))	set
x=frozenset(("apple","banana","cherry"))	frozenset
x=bool(5)	bool
x=bytes(5)	bytes
x=bytearray(5)	bytearray
x=memoryview(bytes(5))	memoryview

（二）Python 的数值类型

Python 中的数值包含 int 和 float 两种类型。为数值型变量赋值时，将创建某种数值类型的变量，如：

```
x=10;       # int
y=6.3;      # float
```

如需查看 Python 中任何对象的类型，可使用 type()函数，如：

```
print(type(x));
print(type(y));
```

1. int

int 是整数类型，是完整的数字，可为正数或负数，不能为小数，长度不限。下面的语句中，前三个语句定义的变量是 int 类型的变量，后面三个语句可用来查看前三个语句定义的变量的数据类型。

```
x=10;
y=37216654545182186317;
z=-465167846;
print(type(x));
print(type(y));
print(type(z));
```

2. Float

Float 是浮点数类型，是包含小数的正数或负数。下面的语句中，前三个语句定义的变

量是 Float 类型的变量，后面三个语句可用来查看前三个语句定义的变量的数据类型。

```
x=3.50;
y=2.0;
z=-63.78;
print(type(x));
print(type(y));
print(type(z));
```

浮点数也可以是科学计数法表示的数，在 Python 中，用 e 表示以 10 为底的幂，如 27e4 表示 27×10^4。使用方法如下例。

```
x=27e4;
y=15E-2;
z=-49.8e100;
print(type(x));
print(type(y));
print(type(z));
```

3. 类型转换

不同的数值类型，可以使用 int()或 float()方法（这里的方法是面向对象程序设计中的概念，指系统已经设计并封装好的操作，用户需要时调用即可）从一种类型转换为另一种类型。如：

```
x=10;   #int
y=6.3;   #float
#把整数转换为浮点数
a=float(x);
#把浮点数转换为整数
b=int(y);
print(a);
print(b);
print(type(a));
print(type(b));
```

第二章　编写程序解决问题的方法

本章的主要内容是介绍程序设计的基本步骤，并通过练习巩固所学知识。本章的目标是通过对算法概念的基本认识，以及对程序流程图的基本了解，让学生掌握分析问题的方法，并按步骤逐步实施，从而达到设计出程序的目的。本章共三节，分别达到掌握程序设计的基本步骤、根据步骤解决简单问题、了解顺序结构程序设计基本结构的教学目标。

第一节　程序设计的基本步骤

一、教材分析

（一）内容分析

本节主要通过让学生按活动步骤，了解从分析问题，到编写出计算机程序代码的过程，在思维和认识上得到提升，明白计算机程序不是计算机自动生成的，而是由人（即程序员）设计完成的。

（二）教学目标

（1）通过教师讲解，掌握流程图的画法。

（2）通过编写、运行交换牛奶和果汁程序的课堂活动，理解算法的作用并能设计算法，了解程序设计的基本步骤。

（三）教学重点和难点

1. 教学重点

（1）程序流程图。

（2）算法的概念。

2. 教学难点

算法概念的理解。

二、学情分析

学生在本节学习之前，对代码程序有一定了解，能录入并运行已经编写好的程序。但对程序是如何编写出来的，还不太清楚，需要通过本节进一步学习。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过回顾前两节的知识要点，对计算机程序设计问题进行宏观思考，分析程序设计与计算机应用软件学习的相同与不同之处，从而引出编程解决问题步骤的本节教学内容。

2. 实践活动，掌握技巧

通过教师讲解和“实践活动”，录入模拟饮料交换的程序并运行。通过程序语句与流程图进行对比，掌握程序代码与程序流程图之间的关系，并能相互转换。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、思考与探索参考答案

1. 思考与探索 1

结合交换饮料的过程想一想，下面的算法能不能交换变量 a 和变量 b 的值，为什么？

①b←a;

②a←b。

解析：

不能交换变量 a 和变量 b 的值。因为 b←a 后，b 的值已改变成 a 的值；再进行 a←b 后，变量 a 和变量 b 的值都是原来的 a 的值。

2. 思考与探索 2

请指出图 2-1-1 所示的流程图中的各框分别与程序中的哪些语句行相对应。

解析：

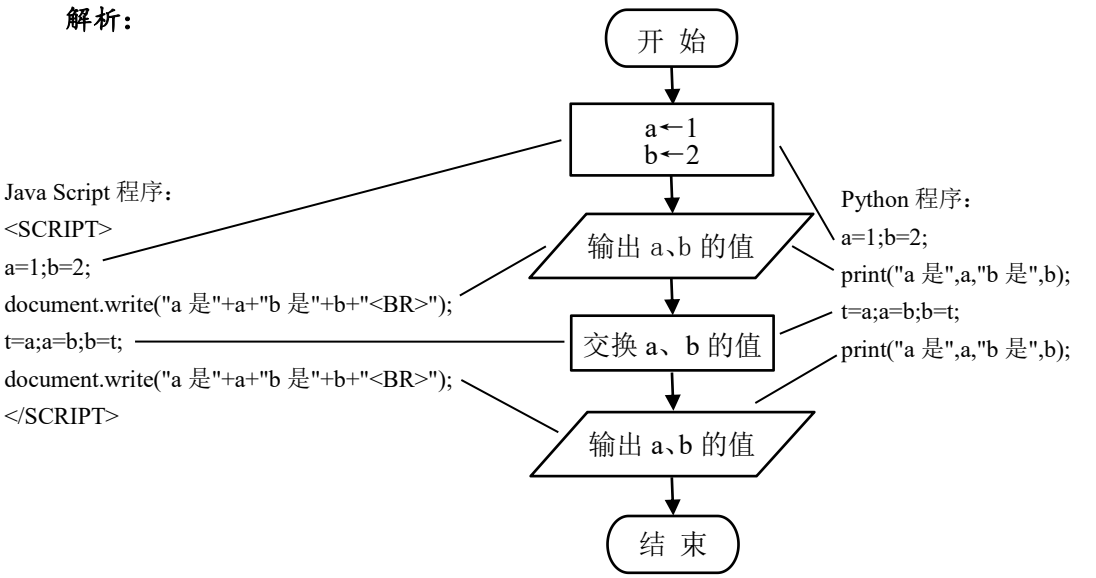

图 2-1-1　交换饮料问题流程图

五、练习提升参考答案

第 1 题：程序设计的基本步骤是什么？

解析：

提出问题→建立数学模型→确定算法→画出流程图→编写程序→上机调试

第 2 题：早晨，阿杜起床准备上学，他要穿戴好袜子、裤子、皮鞋、衬衣、领带、夹克、手表、腰带才能出发，请以完善流程图的方式为他制定一个完成穿戴过程的算法。

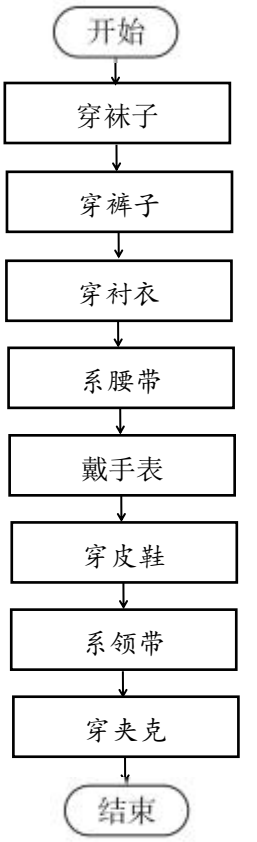

图 2-1-2　流程图

解析：

完善后的流程图如图 2-1-2 所示。

第 3 题：判断正误。

解析：

（1）算法就是计算的方法。（ 错误 ）

（2）算法必须有输出。（ 正确 ）

（3）算法必须在计算机上用某种语言实现。（ 错误 ）

（4）算法不一定有输入。（ 正确 ）

（5）算法必须在执行有限步骤后结束。（ 正确 ）

第二节　简单的程序设计

一、教材分析

（一）内容分析

本节通过温度数值转换和数字拆解两个“实践活动”，让学生进一步熟练掌握流程图和输入、输出、赋值三个语句的使用，巩固上节所学的知识。

（二）教学目标

（1）通过运行温度转换和数字拆解程序的课堂活动，掌握程序设计的基本方法。

（2）通过运行温度转换和数字拆解程序的课堂活动，熟练掌握输入、输出、赋值语句的使用方法。

（3）通过运行温度转换和数字拆解程序的课堂活动，了解程序的基本框架结构。

（三）教学重点和难点

1. 教学重点

（1）输入、输出、赋值语句的使用方法。

（2）整除和取余的概念和应用。

2. 教学难点

整除和取余的概念和应用。

二、学情分析

本节是学生建立流程图和程序代码之间关系的重要阶段，主要任务仍是依据流程图，编写出源程序。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过回顾上节的知识要点，让学生学会画流程图，并能逐步细化流程图，使其能对应程序语句，从而完成程序设计。

2. 实践活动，掌握技巧

通过“实践活动 1”解决“温度转换”问题，巩固上节内容，进一步体验人与程序（机）进行交互的方法。通过“实践活动 2”解决“数位分解”问题，同时介绍新知识点整除和取余，让学生能够理解并能够综合运用。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、练习提升参考答案

第 1 题：汽车仪表盘上显示的速度单位是 km/h，而速度的国际标准单位是 m/s。编写程序解决：输入以 km/h 为单位表示的速度值，输出其对应的以 m/s 为单位的速度值。

解析：

JavaScript 程序：

```
<SCRIPT>
a=Number(prompt("请输入速度（km/h ）：  "));
s=a*1000/3600;
document.write("速度是"+s+"m/s");
</SCRIPT>
```

Python 程序：

```
a=eval(input("请输入速度（km/h ）：  "));
s=a*1000/3600;
```

```
print("速度是",s,"m/s");
```

第 2 题：依次输入两个正整数，分别表示被除数和除数，输出它们做除法运算后的商（商的整数部分）和余数。

解析：

JavaScript 程序：

```
<SCRIPT>
a=Number(prompt("请输入被除数："));
b=Number(prompt("请输入除数："));
s2=a%b;
s1=(a-s2)/b;
document.write("商是："+s1+" 余数是："+s2);
</SCRIPT>
```

Python 程序：

```
a=eval(input("请输入被除数："));
b=eval(input("请输入除数："));
s2=a%b;
s1=a//b;
print("商是：",s1," 余数是：",s2);
```

五、教师知识延伸

JavaScript 技术参考（三）

JavaScript 的算术运算

处理数值的典型情景是算数。

1. 算术运算符

算术运算符对数值（文字或变量）执行算术运算。

表 2.2.1　算术运算符及其含义

运算符	含　义
+	加法
-	减法
*	乘法

续表

运算符	含　义
**	幂
/	除法
%	系数
++	递增
--	递减

2. 算数运算

典型的算术运算会操作两个数值。

这两个数可以是常量，如：

```
var x=7+8;
```

也可以是变量，如：

```
var x = a + b;
```

还可以是表达式，如：

```
var x=(7+8)*a;
```

3. 操作数

在算术运算中，被操作的数称为操作数。

两个操作数之间执行的运算由运算符定义，如表 2.2.2 所示。

表 2.2.2　两个操作数和运算符

操作数	运算符	操作数
7	+	8

4. 加法

加法运算符（+），加数，如：

```
var x=7;
var y=8;
var z=x+y;
```

5. 减法

减法运算符（-），减数，如：

```
var x=7;
var y=8;
```

```
var z=x-y;
```

6. 乘法

乘法运算符（*），乘数，如：

```
var x=7;
var y=8;
var z=x*y;
```

7. 除法

除法运算符（/），除数，如：

```
var x=7;
var y=2;
var z=x/y;
```

8. 取余

取余运算符（%），返回除法的余数，如：

```
var x=7;
var y=2;
var z=x%y;
```

注释：在算术中，两个整数的除法产生商和余数。在 JavaScript 中，取余运算的结果是算术除法的余数。

9. 递增

递增运算符（++），对数值进行递增，如：

```
var x=7;
x++;
var z=x;
```

10. 递减

递减运算符（--），对数值进行递减，如：

```
var x=7;
x--;
var z=x;
```

11. 幂

取幂运算符（**），求第一个操作数的第二个操作数的幂，如：

```
var x=5;
var z=x**2;                    //结果是 25
```

x**y 产生的结果与 Math.pow(x,y)相同，如：

```
var x = 5;
var z = Math.pow(x,2);     // 结果是 25
```

12. 运算符优先级

运算符优先级（Operator precedence）描述了在算术表达式中所执行操作的顺序。

在数学中，乘法比加减法优先；在 JavaScript 中，乘法（*）和除法（/、%）比加法（+）和减法（-）拥有更高的优先级。如：

```
var x=200+50*2;       //结果是 300
```

同时，能够通过使用括号来改变优先级，如：

```
var x=(200+50)*2;       //结果是 500
```

当使用括号时，括号中的运算符会首先被计算。

当多个运算拥有相同的优先级时（比如，加法和减法），它们的计算是从左向右的。

```
var x=200+50-2;       //结果是 248
```

Python 技术参考（三）

Python 的运算符

运算符用于对变量和值执行操作。

Python 有算术运算符、赋值运算符、比较运算符、逻辑运算符、身份运算符、成员运算符、位运算符。

1. 算术运算符

算术运算符与数值一起使用来执行常见的数学运算：

表 2.2.3　算术运算符

运算符	名称	实例
+	加	x+y
-	减	x-y
*	乘	x*y
/	除	x/y
%	取模	x%y
**	幂	x**y
//	整除（取整除）	x//y

2. 赋值运算符

赋值运算符用于为变量赋值，如表 2.2.4 所示。

表 2.2.4　赋值运算符

运算符	实例	等同于
=	x=5	x=5
+=	x+=3	x=x+3
-=	x-=3	x=x-3
*=	x*=3	x=x*3
/=	x/=3	x=x/3
%=	x%=3	x=x%3
//=	x//=3	x=x//3
**=	x**= 3	x=x**3
&=	x&=3	x=x&3
|=	x|=3	x=x|3
^=	x^=3	x=x^3
>>=	x>>=3	x=x>>3
<<=	x<<=3	x=x<<3

3. 关系运算符

比较运算符用于比较两个值，如表 2.2.5 所示。

表 2.2.5　关系运算符

运算符	名称	实例
==	等于	x==y
!=	不等于	x!=y
>	大于	x>y
<	小于	x<y
>=	大于或等于	x>=y
<=	小于或等于	x<=y

4. 逻辑运算符

逻辑运算符用于组合多个条件，如表 2.2.6 所示。

表 2.2.6　逻辑运算符

运算符	描　　述	实　　例
and	如果两个语句都为真，则返回 True	x>3 and x<10
or	如果其中一个语句为真，则返回 True	x>3 or x<4
not	反转结果，如果结果为 True，则返回 False	not(x>3 and x<10)

5. 身份运算符

身份运算符用于比较对象，不是比较它们是否相等，但如果它们实际上是同一个对象，则具有相同的内存位置。

6. 成员运算符

成员运算符用来表示两个变量是否是同一个对象，如表 2.2.7 所示。

表 2.2.7　成员运算符

运算符	描　述	实　例
is	如果两个变量是同一个对象，则返回 True	x is y
is not	如果两个变量不是同一个对象，则返回 True	x is not y

7. 位运算符

位运算符用于比较（二进制）数字，如表 2.2.8 所示。

表 2.2.8　位运算符

运算符	描　述	实　　例
&	AND	如果两个位均为 1，则将每个位设为 1
|	OR	如果两位中的一位为 1，则将每个位设为 1
^	XOR	如果两个位中只有一位为 1，则将每个位设为 1
~	NOT	反转所有位
<<	Zero fill left shift	通过从右侧推入零来向左移动，推掉最左边的位
>>	Signed right shift	通过从左侧推入最左边的位的副本向右移动，推掉最右边的位

续表

运算符	描　述	实　　例
|	OR	如果两位中的一位为 1，则将每个位设为 1
^	XOR	如果两个位中只有一位为 1，则将每个位设为 1
~	NOT	反转所有位
<<	Zero fill left shift	通过从右侧推入零来向左移动，推掉最左边的位
>>	Signed right shift	通过从左侧推入最左边的位的副本向右移动，推掉最右边的位

第三节　计算机与计算器
——顺序程序设计

一、教材分析

（一）内容分析

本节通过两个让计算机进行数学计算的程序，让学生认识到同样进行数学计算，计算机与计算器是有本质区别的，进一步明白什么情况下用计算器解决问题，什么时候用程序解决问题。通过“实践活动 2”，让学生具备依据流程图编写程序的能力。

（二）教学目标

（1）通过编写、运行计算总分、平均分，以及计算圆柱体表面积程序的课堂活动，掌握顺序结构程序设计的基本方法。

（2）通过编写、运行计算总分、平均分，以及计算圆柱体表面积程序的课堂活动，理

解计算机和计算器的本质区别。

（3）通过编写、运行计算总分、平均分，以及计算圆柱体表面积程序的课堂活动，进一步理解程序设计的步骤。

（三）教学重点和难点

1. 教学重点

（1）根据流程图编写程序。

（2）计算机和计算器的本质区别。

2. 教学难点

计算机和计算器的本质区别。

二、学情分析

学生在学习本节之前，明白了编写程序的基本步骤，但实现能力还有待提高。本节的实践活动目的在于提高学生这方面的能力。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过回顾前两节的知识要点，让学生体会程序语句执行的先后顺序，并通过流程图理解程序的执行顺序，从而引入程序结构的概念，并明白顺序结构程序设计。

2. 实践活动，掌握技巧

通过“实践活动 1”解决“分数计算”问题，巩固本章内容，进一步体验程序的计算功能。通过“实践活动 2”解决“圆柱体表面积计算”问题，体验根据流程图编写程序的方法。

3. 总结要点，扩展知识

根据学生“实践活动”的完成情况，进行要点总结。

四、练习提升参考答案

第 1 题：编写程序，输入四门功课的成绩，输出总成绩和平均成绩。

当输入四门功课成绩分别为 132、146、98、101 时，运行结果为（　　）。

A．总分：477　平均分：119.25　　B．总分：0　　平均分：0

C．总分：467　平均分：119.25　　D．总分：477　平均分：119

解析：

答案：C

第 2 题：已知圆的半径为 128，编写程序，计算并输出圆的周长和面积。

（1）程序运行结果周长为（　　）。（圆周率取 3.14）

A．0　　B．160.76　　C．80.38　　D．803.84

解析：

答案：D

（2）程序运行结果面积值为（　　）。（圆周率取 3.14）

A．0　　B．1289.16　　C．51445.58　　D．51445.76

解析：

答案：D

第 3 题：甲、乙、丙分别有 36 本、48 本、64 本书。首先由甲把他的书平分为三份，分给乙和丙各一份，自己留一份；然后乙和丙按与甲相同的方法处理。编写程序，模拟上述过程，并输出所有中间结果。

最后甲、乙、丙各有多少本书？

A．甲：36　乙：48　丙：64　　B．甲：64　乙：52　丙：32

C．甲：12　乙：60　丙：76　　D．甲：36　乙：18　丙：24

解析：

答案：A

第三章 认识计算机的逻辑判断能力

本章的主要内容是介绍分支程序设计，并通过练习巩固认知。结合前面讲的顺序程序设计，会利用程序的分支语句，解决实际生活中的相关问题。本章共三节，分别达到：掌握分支程序设计的基本流程、理解逻辑运算及其运用、了解分支嵌套程序设计结构。

第一节 程序也会见机行事
——分支程序设计

一、教材分析

（一）内容分析

本节重点讲解分支程序设计的基本结构和流程。通过让学生按步骤实施“实践活动”，了解什么是分支问题及程序处理分支问题的方法，从而引出 if 语句的两种常用方式，并让学生掌握运用分支语句编写程序的方法。

（二）教学目标

（1）通过实践活动，理解分支结构的意义。

（2）通过总结讲解并结合流程图，从本质上理解分支程序设计思想及应用，并掌握分

支语句及其执行过程。

（三）教学重点和难点

1. 教学重点

（1）if 语句的书写格式。

（2）分支程序设计思想。

2. 教学难点

两种条件语句使用方式的联系与区别。

二、学情分析

学生在本节学习之前，能录入并运行已经编写好的程序，对顺序结构的程序有一定了解，能编写简单的顺序结构程序，但分支问题如何解决，分支程序如何编写，还需要通过本节进一步学习。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过运行一个计算算术平方根的程序，根据数学中负数没有算术平方根这个特点，让学生体验程序运行时出错的现象，并体会计算机是如何处理程序中的错误的。为了避免程序出现错误，需要程序对数据进行判断，并根据判断结果进行不同的处理，从而引入分支程序设计。

2. 实践活动，掌握技巧

通过“实践活动 1”解决“平方根计算”问题，通过“思考与探索”明白分支程序设计的必要性。根据分支程序流程图，讲解 if 语句的执行流程。巩固本章内容，进一步体验程序的计算功能。通过“实践活动 2”解决“圆柱体表面积计算”问题，体验根据流程图编写程序的方法。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、思考与探索参考答案

根据数学知识可知负数没有平方根，如果输入负数会有什么结果呢？

解析：

这时，会提示程序出错。其中：

NaN（Not a Number，非数）是计算机科学中数值数据类型的一类值，表示未定义或不可表示的值。

ValueError：math domain error，表示数学运算错误。

四、练习提升参考答案

第 1 题：依据图 3-1-6 所示的流程图编写程序，输入一个数，输出该数的绝对值。

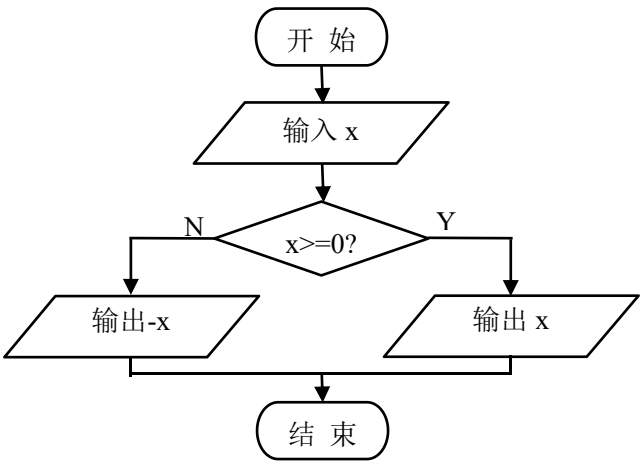

图 3-1-6　输出绝对值程序流程图

解析：

根据流程图可以编写如下程序。

JavaScript 程序：

```
<SCRIPT>
x=prompt("x=");            //输入数据存入变量 x
if (x>=0) alert(x); else alert(-x);
</SCRIPT>
```

Python 程序：

```
x=eval(input("x="));          #输入数据存入变量 x
if x>=0:
    print(x);
```

```
else:
    print(-x);
```

第 2 题：某公司为学生制作胸卡，制作 200 张和 200 张以上时，按每张 3 元计价，否则按每张 4 元计价。依据图 3-1-7 所示的流程图编写程序，实现根据输入的胸卡数输出应付的总金额。

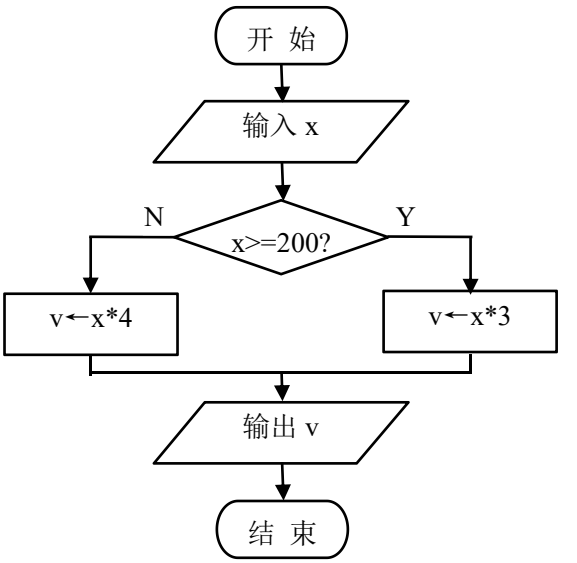

图 3-1-7　计算应付金额程序流程图

解析：

根据流程图可以编写如下程序。

JavaScript 程序：

```
<SCRIPT>
x=prompt("x=");            //输入数据存入变量 x
if (x>=200) alert(x*3); else alert(x*4);
</SCRIPT>
```

Python 程序：

```
x=eval(input("x="));           #输入数据存入变量 x
if x>=200:
    print(3*x);
else:
    print(4*x);
```

第二节　程序也有逻辑思想
——条件复杂的分支程序设计

一、教材分析

（一）内容分析

本节通过实践活动，让学生了解较为复杂条件的表达方式，学会利用逻辑运算表示复杂条件，并进一步熟练掌握流程图和条件语句的使用，巩固上节的内容。

（二）教学目标

（1）通过实践活动，掌握分支结构的执行过程。

（2）通过实践活动，理解条件语句中复杂条件的表示及应用。

（3）通过实践活动，掌握逻辑运算的运算规则及应用。

（三）教学重点和难点

1. 教学重点

（1）分支结构的执行过程。

（2）条件语句中复杂条件的表示及应用。

2. 教学难点

逻辑运算的运算规则及应用。

二、学情分析

本节是学生进一步理解、掌握分支程序设计的重要阶段，主要任务是通过实践活动，了解逻辑运算在处理复杂条件时的必要性，并掌握其应用规则。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”

“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过回顾上节内容，并对“算术平方根计算”问题进行改造，使判断条件变得复杂，从而引入新知识点——逻辑运算。

2. 实践活动，掌握技巧

通过“实践活动”——“求$\sqrt{1-x^2}$的值”问题，讲解关系运算，引出逻辑运算并进行讲解。通过“思考与探索”进一步理解逻辑运算并能综合运用。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、思考与探索参考答案

某次实践活动对参与选手进行智力、生活两项技能测试和一项操作实践测试。规定：对两项技能测试成绩均在 85 分以上或总分在 270 分以上的选手给予奖励。编写程序根据输入的三项测试成绩决定是否获得奖励。

解析：

略。

五、练习提升参考答案

第 1 题：如果把前面“思考与探索”中的问题奖励条件改为“一项成绩为 100 分，其余两项成绩均大于 60 分”，程序该如何修改？

解析：

奖励条件可以用多种逻辑表达式来表达，下面是其中比较简洁的一种。

JavaScript 程序：

```
<SCRIPT>
   n1=Number(prompt("输入智力技能测试成绩："));
   n2=Number(prompt("输入生活技能测试成绩："));
   n3=Number(prompt("输入操作实践成绩："));
   n=n1+n2+n3;   //计算总分
   if ((n1==100||n2==100||n3==100)&&(n1>=60&&n2>=60&&n3>=60))
      alert("奖励");
```

```
</SCRIPT>
```

Python 程序：

```
n1=eval(input("输入智力技能测试成绩："));
n2=eval(input("输入生活技能测试成绩："));
n3=eval(input("输入操作实践成绩："));
if (n1==100 or n2==100 or n3==100 ) and (n1>=60 and n2>=60 and n3>=60):
    print("奖励");
```

第 2 题：输入年份，输出该年份是否为闰年。

提示：满足下列条件之一的年份为闰年。

（1）能被 4 整除，但不能被 100 整除。

（2）能被 400 整除。

解析：

JavaScript 程序：

```
<SCRIPT>
  n=Number(prompt("请输入年份："));
  if ((n%400==0||n%4==0&&n%100!=0))
    alert("是闰年");
  else
    alert("不是闰年");
</SCRIPT>
```

Python 程序：

```
n=eval(input("请输入年份："));
if ((n%400==0 or n%4==0 and n%100!=0)):
    print("是闰年");
else:
    print("不是闰年");
```

六、教师知识延伸

JavaScript 技术参考（四）

If...Else 语句

条件语句用于基于不同条件执行不同的动作。

1. 条件语句

在编写程序时，经常需要基于不同判断执行不同的动作，这时可以在代码中使用条件语句来实现。其完整格式是：

```
if <条件>
   <代码块 1>
else
   <代码块 2>
……
```

其中，<代码块 1>是 if 语句指定的条件成立（值为 true）时，执行的代码块；<代码块 2>是 if 语句指定的条件不成立（值为 false）时，执行的代码块。

在 JavaScript 中，可使用如下几种形式的条件语句。

（1）当 if 语句指定的条件为 true 时，执行<代码块 1>；反之，执行 if 语句后面的语句。格式为：

```
if <条件>
   <代码块 1>
……
```

（2）在 if 语句中，使用 else 规定条件为 false 时要执行的语句组，即<代码块 2>。格式为：

```
if <条件>
   <代码块 1>
else
   <代码块 2>
……
```

（3）在 if 语句中，使用 else if 来规定条件为 false 时要测试的新条件。格式为：

```
if <条件 1>
   <代码块 1>
else if <条件 2>
        <代码块 2>
    else
        <代码块 3>
……
```

（4）在 if 语句中，使用 switch 规定多个被执行的备选语句组。

2. if 语句

使用 if 语句可规定条件为 true 时被执行的 JavaScript 代码块，如：

```
if (条件) {
    如果条件为 true 时执行的代码
}
```

注意：if 使用小写字母。大写字母（IF 或 If）会产生 JavaScript 错误。

【例 1】hour 表示时间，如果时间早于 18:00，即 hour 小于 18，则问候语赋值为"Good day"。程序为：

```
if (hour<18) {
    greeting="Good day";
}
```

如果时间早于 18:00，则 greeting 的结果将是 Good day。

3. else 语句

使用 else 语句来规定条件为 false 时的代码块，如：

```
if (条件) {
    条件为 true 时执行的代码块
} else {
    条件为 false 时执行的代码块
}
```

【例 2】如果 hour 小于，创建"Good day"问候语；否则，创建"Good evening"问候语。程序为：

```
if (hour<18) {
    greeting = "Good day";
  } else {
    greeting="Good evening";
  }
```

如果 hour 为 20，则 greeting 的结果为 Good evening。

4. else if 语句

使用 else if 来规定当前面条件为 false 时的新条件，如：

```
if (条件 1) {
    条件 1 为 true 时执行的代码块
} else if (条件 2) {
    条件 1 为 false 而条件 2 为 true 时执行的代码块
  } else {
    条件 1 和条件 2 同时为 false 时执行的代码块
}
```

【例 3】如果时间早于 10:00，则创建"Good morning"问候；否则，如果时间晚于 10:00 但早于 18:00，则创建"Good day"问候，否则，创建"Good evening"问候。程序为：

```
if (time<10) {
    greeting="Good morning";
  } else if (time < 18) {
    greeting="Good day";
  } else {
    greeting="Good evening";
  }
```

时间为 20:00 时，greeting 的结果为 Good evening。

Python 技术参考（四）

Python If...Else

1. Python 条件和 If 语句

Python 的关系运算及由其构成的关系式举例，如表 3.3.1。

表 3.3.1　关系运算及由其构成的关系式举例

关系运算名	等于	不等于	小于	小于等于	大于	大于等于
关系运算符	==	!=	<	<=	>	>=
关系式举例	a==b	a!=b	a<b	a<=b	a>b	A>=b

这些关系式可构成 Python 的条件，并能够以多种方式使用，最常见的是 if 语句和循环。

if 语句使用 if 关键词来写，如：

```
a=66;
b=200;
if b>a:
   print("b is greater than a");
```

在这个例子中，我们使用了两个变量 a 和 b，作为 if 语句的一部分，它们用于测试 b 是否大于 a。因为 a 是 66，而 b 是 200，if 语句中的条件成立，所以将“b is greater than a”打印到屏幕。

2. 缩进

Python 依赖缩进，使用空格来定义代码中的范围。其他编程语言通常使用花括号来实现此目的。

没有缩进的 if 语句（会引发错误），如：

```
a=66;
```

```
b=200;
if b>a:
print("b is greater than a"); # 会报错
```

3. Elif

elif 关键字是 Python 对“如果之前的条件不正确，那么试试这个条件”的表达方式，如：

```
a=66;
b=66;
if b>a:
   print("b is greater than a");
elif a==b:
   print("a and b are equal");
```

在这个例子中，a 等于 b，所以第一个条件不成立，但 elif 条件为 true，所以输出“a and b are equal”。

4. Else

else 关键字捕获未被之前的条件捕获的任何内容，如：

```
a=200;
b=66;
if b>a:
   print("b is greater than a");
elif a==b:
   print("a and b are equal");
else:
   print("a is greater than b");
```

在这个例子中，a 大于 b，所以第一个条件不成立，elif 条件也不成立，所以转到 else 条件并输出“a is greater than b”。

也可以使用没有 elif 的 else，如：

```
a=200;
b=66;
if b>a:
   print("b is greater than a");
else:
   print("b is not greater than a");
```

5. 简写 If

如果只有一条语句要执行，则可以将其与 if 语句放在同一行。

单行 if 语句，如：

```
a 200;
b=66;
if a>b:print("a is greater than b");
```

6. 简写 If...Else

如果只有两个语句要执行，一个用于 if，另一条用于 else，则可以将它们全部放在同一行。单行 if else 语句，如：

```
a=200;
b=66;
print("A") if a > b else print("B")
```

还可以在同一行上使用多个 else 语句。单行 if else 语句，有三个条件，如：

```
a=200;
b=66;
print("A") if a>b else print("=") if a==b else print("B")
```

7. and

and 关键字是一个逻辑运算符，用于组合条件。

测试 a 是否大于 b，且 c 是否大于 a。程序如下：

```
a=200;
b=66;
c=500;
if a>b and c>a:
  print("Both conditions are True");
```

8. or

or 关键字也是逻辑运算符，用于组合条件语句。

测试 a 是否大于 b，或者 a 是否大于 c。程序如下：

```
a=200;
b=66;
c=500;
if a>b or a>c:
  print("At least one of the conditions is True");
```

9. 嵌套 If

可以在 if 语句中包含 if 语句，这称为嵌套 if 语句，如：

```
x=52;
if x>10:
   print("Above ten,");
   if x>20:
      print("and also above 20!");
   else:
      print("but not above 20.");
```

10. pass 语句

if 语句不能为空，但是如果出于某种原因写了无内容的 if 语句，可使用 pass 语句来避免错误，如：

```
a=66;
b=200;
if b>a:
   pass;
```

第三节　程序也会多种抉择
——复杂的分支程序设计

一、教材分析

（一）内容分析

分支可以用条件语句来实现，本节讲述两种以上的多分支的实现方法。多分支可以通过分支条件去处理，也可以用分支嵌套来处理。本节通过两个实例，让学生了解多分支的处理方法，并能够编程实现。

（二）教学目标

（1）通过两个“实践活动”——用不同方式实现“写评语”程序，让学生分析问题，掌握分支嵌套的执行过程。

（2）通过两个“实践活动”——用不同方式实现“写评语”程序，让学生分析问题，理解分支嵌套的设计思想及应用。

（三）教学重点和难点

1. 教学重点

（1）分支嵌套的执行过程。

（2）分支嵌套的设计思想及应用。

2. 教学难点

分支嵌套的设计思想及应用。

二、学情分析

学生在本节学习之前，已经明白分支程序设计的基本思想，但在实际问题中，条件常常是比较复杂的，需要通过较为复杂的分支嵌套方式才能实现。解决本节的问题有一定难度，学生可能会接受困难，只要认真分析，就可以理解。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过回顾分支程序内容，并对“成绩等级”问题进行分析，掌握多分支程序的一种处理方法——分支嵌套。

2. 实践活动，掌握技巧

通过两个“实践活动”——用不同方式实现“写评语”程序，在两种方法的对比中深入理解多分支问题的解决方法，使学生达到善于思考，并能够综合运用的目的。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、实践活动指导

在刚进行的数学测验中，根据输入的某位同学的成绩输出相应的评语：如果成绩在 85 分及以上，输出评语“优秀”；如果成绩在 75 分至 85 分之间，输出评语“良好”；如果成绩在 60 分至 75 分之间，输出评语“中等”；如果成绩不足 60 分，输出评语“需努力”。

解析：

“实践活动 1”和“实践活动 2”都是解决这一问题的程序，其中，“实践活动 1”的程序使用了分支的嵌套，“实践活动 2”中的程序没有使用分支的嵌套。

可以发现没有分支嵌套程序的运行结果与前面有分支嵌套的程序的运行结果相同，能起到同样的判断效果。哪一个程序执行的语句更少呢？从效率和程序可读性两方面对比两种程序的优点与不足。

分支嵌套程序的优点：

程序执行语句更少，程序执行效率高。本问题中条件语句的条件跟问题要求比较一致，是推荐的编程方法。

没有分支嵌套程序的优点与不足：

程序可读性强，但条件设置会比问题描述复杂，需要再处理。

五、练习提升参考答案

第 1 题：设 x 为货物的质量，y 为火车托运费，运费与货物质量的关系为：

$$y=\begin{cases}0.2x & 0<x\leqslant 20\\ 0.2x+0.3(x-20) & 20<x\leqslant 100\end{cases}$$

货物的质量不能为负数或 0，也不能超过 100，为负数或超过 100 均属于不能托运的质量。

编程解决该问题的流程图，如图 3-3-4 所示。

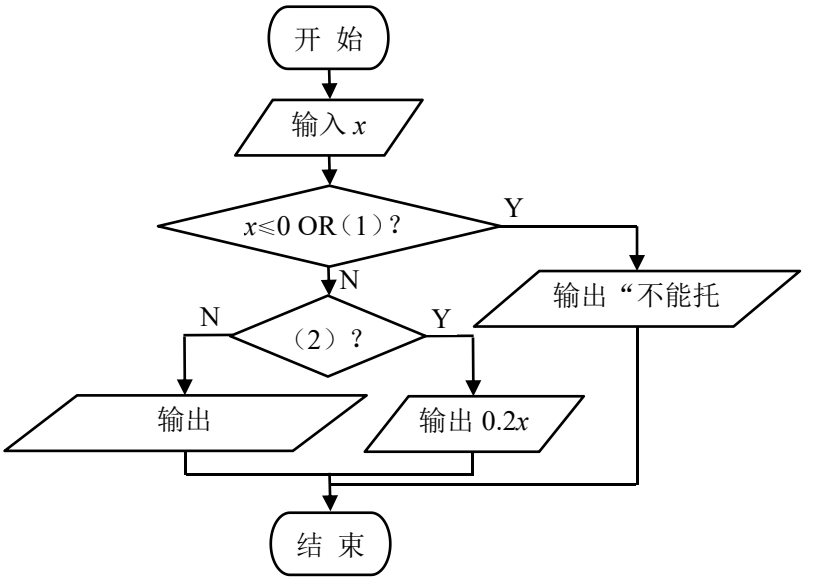

图 3-3-4　运费程序流程图

解析：

流程图中，在（1）框处应填的选项是__A__。

A. $x>20$　　B. $x>0$　　C. $20<x\leqslant 100$　　D. $0<x\leqslant 20$

流程图中，在（2）框处应填的选项是__B__。

A. $x\leqslant 100$　　B. $x\leqslant 20$　　C. $0<x<20$　　D. $20<x\leqslant 100$

JavaScript 程序：

```
<SCRIPT>
   x=Number(prompt("输入货物质量："));
   if (x<=0||x>100)   alert("不能运送");
   if (x<=20)
        alert(0.2*x);
   else alert(0.2*x+0.3*(x-20));
</SCRIPT>
```

Python 程序：

```
x=eval(input("输入货物质量："));
if (x<=0 or x>100):
      print("不能运送");
else:
    if (x<=20):
         print(0.2*x);
```

```
    else:
        print(0.2*x+0.3*(x-20));
```

第 2 题：运输问题。

设 *s* 表示运输路程（单位：千米），*w* 表示货物质量（单位：吨），每吨货物每千米的运费价格（单位：元）如表 3-3-1 所示，计算费用时采用分段计算（标准见表 3-3-1）。编写程序：输入 *s* 和 *w* 的值，计算并输出运费金额。

表 3-3-1　某运输公司的运输收费标准

路程	*s*<250	250≤*s*<500	500≤*s*<1000	1000≤*s*<2000	2000≤*s*<3000	*s*≥3000
单价	100	98	95	90	80	70

解析：

JavaScript 程序：

```
<SCRIPT>
   s=Number(prompt("输入运输路程："));
   w=Number(prompt("输入货物质量："));
   if (s<250) p=100;
   if (s>=250&&s<500) p=98;
   if (s>=500&&s<1000) p=95;
   if (s>=1000&&s<2000) p=90;
   if (s>=2000&&s<3000) p=80;
   if (s>=3000) p=70;
   alert("运输费用："+p*w);
</SCRIPT>
```

Python 程序：

```
s=eval(input("输入运输路程："));
w=eval(input("输入货物重量："));
if (s<250):p=100;
if (s>=250 and s<500):p=98;
if (s>=500 and s<1000):p=95;
if (s>=1000 and s<2000):p=90;
if (s>=2000 and s<3000):p=80;
if (s>=3000):p=70;
print("运输费用：",p*w);
```

第四章　认识计算机的速度

本章的主要内容是介绍循环程序设计，并通过练习巩固认识。结合前面讲的顺序程序设计、分支程序设计，会利用程序的循环语句，解决实际生活中的相关问题。本章共五节，目标是：掌握循环程序设计的基本流程、理解穷举法、了解文件输入输出，为提交答案类型问题的解决提供基础。

第一节　感受计算机的速度
——循环程序设计

一、教材分析

（一）内容分析

本节重点讲解循环程序设计的基本结构及其执行流程，通过让学生按“实践活动”的步骤实践操作，对比是否用循环结构对重复执行的程序进行编程的方便性、可读性的影响，了解循环的概念、循环的结构及其执行流程，初步掌握循环程序的编程方法。

（二）教学目标

（1）通过“实践活动”和总结讲解，掌握循环语句的执行过程和使用方法。

（2）通过流程图加深对循环结构及其执行过程的理解，从本质上理解循环程序设计思想及其应用。

（三）教学重点和难点

1. 教学重点

（1）认识循环语句及书写格式。

（2）循环程序设计思想。

2. 教学难点

理解循环语句。

二、学情分析

学生在本节学习之前，能录入并运行已经编写好的程序，并对顺序结构、分支结构的程序有一定了解，能编写简单的顺序结构程序、分支结构程序，但对于循环问题如何解决，循环程序如何编写，还需要通过本节进一步学习。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过运行几个不同数据规模的“算术平方根计算”程序，让学生感受计算机的速度。这在其他应用软件中不容易感觉到，学生应该具有比较深刻的印象，从而对程序代码的编写更感兴趣。

2. 实践活动，掌握技巧

通过“实践活动”感受计算机的速度，再通过程序流程图了解影响程序执行快慢的原因，了解循环程序代码的书写格式。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。如果学生对本节知识接受良好，可以对“阅读拓展”中的内容进行扩展介绍。

四、练习提升参考答案

第 1 题：编程输出 1—100 中所有偶数的算术平方根。

解析：

JavaScript 程序：

```
<SCRIPT>
for (i=2;i<=100;i+=2)
document.write(Math.sqrt(i)+"<BR>");
</SCRIPT>
```

Python 程序：

```
import math
for i in range(2,101,2):
    print(math.sqrt(i));
```

第 2 题：编程输出 1—100 中所有奇数的算术平方根。

解析：

JavaScript 程序：

```
<SCRIPT>
for (i=1;i<=100;i+=2)
document.write(Math.sqrt(i)+"<BR>");
</SCRIPT>
```

Python 程序：

```
import math
for i in range(1,101,2):
    print(math.sqrt(i));
```

第 3 题：如图 4-1-2 所示的流程图中，循环部分是_____。

A.（1）（2）（3）（4）　　B.（2）（3）（4）

C.（3）（4）　　D.（1）（3）（4）

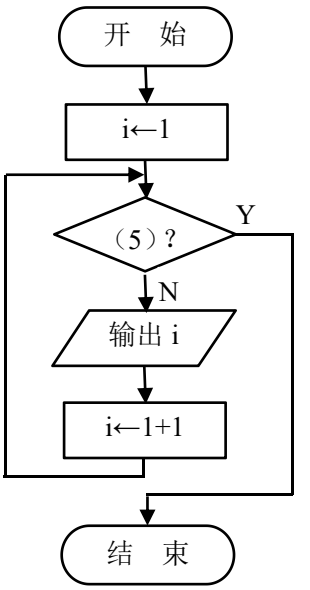

图 4-1-2　流程图

解析：

答案：B

4.下面程序运行的结果为____。

JavaScript 程序：

```
<SCRIPT>
   for (i=1;i<=9;i=i+3)
      document.write(i+"<br>");
</SCRIPT>
```

Python 程序：

```
for i in range(1,10,3):
    print(i);
```

A. 1 4 7　　B. 1 3 6　　C. 1 4 7 10　　D. 1 3 6 9

解析：

答案：A

六、教师知识延伸

JavaScript 技术参考（五）

（一）JavaScript 循环

循环可多次执行代码块。

如果需要多次运行同一段代码，且每次使用不同的值，那么使用循环相当方便。

例如：遇到下面使用数组的例子。

不使用循环，可以写出如下代码：

```
text+=cars[0]+"<br>";
text+=cars[1]+"<br>";
text+=cars[2]+"<br>";
text+=cars[3]+"<br>";
text+=cars[4]+"<br>";
text+=cars[5]+"<br>";
```

使用循环，可写出如下代码：

```
for (i=0;i<cars.length;i++) {
      text+=cars[i]+"<br>";
 }
```

显然，使用循环比不使用循要简洁、方便，写出的代码少，结构清晰，可读性强。

（二）不同类型的循环

JavaScript 支持下列类型的循环：

（1）for：多次遍历代码块；

（2）for/in：遍历对象属性；

（3）while：当指定条件为 true 时循环执行一段代码块；

（4）do/while：执行一段代码块，再判断条件，当指定条件为 true 时循环执行代码块。

（三）For 循环

for 循环是创建循环时经常使用的循环类型。

for 循环的格式如下：

```
for (语句 1;语句 2;语句 3) {
      要执行的代码块
```

```
}
```

语句 1 在循环（代码块）开始之前执行。

语句 2 定义执行循环（代码块）的条件。

语句 3 在循环体（要执行的代码块）每次被执行后执行。

【例 1】

```
for (i=0;i<5;i++) {
        text += "数字是  "+i+"<br>";
}
```

从上面的代码中，可以看到：

语句 1 在循环开始之前设置了一个变量（var i=0）；

语句 2 定义运行循环的条件（i<5）。

语句 3 在每次执行代码块之后对变量 i 的值进行递增（i++）。

（1）语句 1。通常使用语句 1 来初始化循环中所使用的变量（i=0），但语句 1 是可选的。

可以在语句 1 中初始化多个值（由逗号分隔），如：

```
for (i=0,len=cars.length,text="";i<len;i++) {
        text+=cars[i]+"<br>";
}
```

如果在循环开始前设置好了初始值，还可以省略语句 1，如：

```
var i=2;
var len=cars.length;
var text="";
for (;i<len;i++) {
        text+=cars[i]+"<br>";
}
```

（2）语句 2。语句 2 通常用于设置循环变量的条件，但语句 2 也是可选的。

如果语句 2 返回 true，那么循环会继续开始；如果返回 false，则循环将结束。

如果省略语句 2，那么必须在循环中提供一个 break；否则，循环永远不会结束。

（3）语句 3。通常语句 3 会递增循环变量的值，但语句 3 也是可选的。

语句 3 可做任何事情，比如：负递增（i--）、正递增（i=i+15），或者任何其他事情。

如果在循环体内递增循环变量的值时，语句 3 也可被省略，如：

```
var i=0;
var len=cars.length;
for (;i<len;) {
```

```
        text+=cars[i]+"<br>";
          i++;
    }
```

（四）While 循环

1. While 循环

只要指定的条件为 true，while 循环就会一直循环执行代码块。

格式如下：

```
while (条件) {
    要执行的代码块
}
```

【例 2】在下面的例子中，只要变量（i）小于 10，循环中的代码将一直执行。

```
while (i < 10) {
    text+="数字是 "+i;
    i++;
}
```

如果没有对条件中的变量进行递增，那么循环永不会结束。

2. Do/While 循环

do/while 循环是 while 循环的变体。在检查条件是否为真之前，这种循环会执行一次代码块，然后只要条件为真就会重复循环。

格式如下：

```
do {
    要执行的代码块
}
while (条件);
```

【例 3】下面的例子使用了 do/while 循环。即使条件为 false，该循环也会执行一次，因为代码块会在条件测试之前执行。

```
do {
    text+="The number is "+i;
    i++;
  }
while (i<10);
```

不要忘记对条件中所用变量进行递增，否则循环永不会结束！

（五）比较 For 与 While

while 循环与 for 循环相当类似，其中的语句 1 和语句 2 都可以省略。

【例 4】下例中的循环使用 for 循环来提取 cars 数组中的汽车品牌。

```
var cars=["比亚迪","长城","长安","奇瑞"];
var i=0;
var text="";
for (;cars[i];) {
      text+=cars[i]+"<br>";
      i++;
}
```

【例 5】下例中的循环使用 while 循环来提取 cars 数组中的汽车品牌。

```
var cars="比亚迪","长城","长安","奇瑞"];
var i=0;
var text="";
while (cars[i]) {
      text+=cars[i]+"<br>";
      i++;
}
```

Python 技术参考（五）

（一）for 循环

for 循环用于迭代序列（即列表、元组、字典、集合或字符串）。

Python 中的 For 循环与其他编程语言中的 for 关键字不太一样，而是更像其他面向对象编程语言中的迭代器方法。

1. 为列表、元组、集合中的每个项目等执行一组语句

【例 6】下例，打印 fruits 列表中的每种水果。

```
fruits=["苹果","香蕉","樱桃"];
for x in fruits:
   print(x);
```

提示：for 循环不需要预先设置索引变量。

2. 循环遍历字符串

for 循环甚至可以让字符串作为可迭代的对象。

【例 7】遍历单词“banana”中的字母。

```
for x in "banana":
  print(x);
```

3. break 语句

通过使用 break 语句，可以在循环遍历所有项目之前停止循环。

【例 8】如果 x 是"香蕉"，则退出循环。

```
fruits=["苹果","香蕉","樱桃"];
for x in fruits:
  print(x);
  if x=="香蕉":
    break;
```

【例 9】当 x 为"香蕉"时退出循环，要求在打印之前中断。

```
fruits=["苹果","香蕉","樱桃"];
for x in fruits:
  if x=="香蕉":
    break;
  print(x);
```

4. continue 语句

continue 语句可以停止循环的当前迭代，并继续下一个迭代。

【例 10】不输出香蕉。

```
fruits=["苹果","香蕉","樱桃"];
for x in fruits:
  if x=="香蕉":
    continue;
  print(x);
```

5. range() 函数

如需循环一组代码指定的次数，可以使用 range()函数。

range()函数返回一个数字序列，默认情况下从 0 开始，递增 1（默认地），并以指定的数字结束。

【例 11】使用 range()函数。

```
for x in range(10):
  print(x);
```

注意：range(10)不是 0 到 10 的值，而是值 0 到 9 的值。

range()函数默认 0 为起始值，不过可以通过添加参数来指定起始值，如 range(3,10)意味着值为 3 到 10（但不包括 10）。

【例 12】使用起始参数。

```
for x in range(3,10):
  print(x);
```

range()函数默认将序列递增 1，但是可以通过添加第三个参数来指定增量值，如 range(2,30,3)表示：值为 2 到 30（但不包括 30），每次递增 3。

【例 13】使用 6 递增序列（默认值为 1）。

```
for x in range(3,50,6):
  print(x);
```

6. For 循环中的 Else

for 循环中的 else 关键字指定循环结束时要执行的代码块。

【例 14】打印 0 到 9 的所有数字，并在循环结束时输出一条消息。

```
for x in range(10):
  print(x);
else:
  print("Finally finished!");
```

7. 嵌套循环

嵌套循环是循环内的循环。

“外循环”每迭代一次，“内循环”将执行一遍。

【例 15】输出每个水果的每个形容词。

```
adj=["红","大","味美"]
fruits=["苹果","香蕉","樱桃"];
for x in adj:
  for y in fruits:
    print(x, y);
```

8. pass 语句

for 语句不能为空，但是如果由于某种原因写了无内容的 for 语句，可使用 pass 语句来避免错误。如：

```
for x in [0, 1, 2]:
  pass;
```

（二）While 循环

使用 while 循环，只要条件为真就可以执行一次循环体语句。

【例 16】只要 i 小于 7，输出 i。

```
i=1
while i<7:
   print(i);
   i+=1;
```

注释：一定要递增 i，否则循环会永远继续。

while 循环要准备好相关的变量。在上例中，定义了一个索引变量 i，并将其设置为 1。

在 while 循环中，可以使用 break 语句、continue 语句、else 语句，它们的含义和用法与 for 循环一致。

第二节　计算数列的和与积
——累加与累乘

一、教材分析

（一）内容分析

本节通过实践活动，让学生了解利用程序结构实现算法，学会利用循环实现累加与累乘两种算法，并进一步加强通过程序实现算法思想的方法。

（二）教学目标

（1）通过实践活动，掌握循环结构的执行过程。

（2）通过实践活动，掌握累加与累乘算法。

（三）教学重点和难点

1. 教学重点

（1）循环结构的执行过程。

（2）累加与累乘算法。

2. 教学难点

循环程序实现算法思想。

二、学情分析

本节是学生进一步理解并掌握循环程序设计的重要阶段，主要任务是通过实践活动了解累加与累乘算法，并掌握其应用规则。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过回顾上节循环程序知识要点，引入“数学计算问题”，让学生明白尽管程序是计算机执行的，但方法是编写程序的人设计出的，由想出方法到设计算法，从而对程序代码的编写更感兴趣。

2. 实践活动，掌握技巧

通过“实践活动”体验程序的高超的计算能力，再通过流程图了解程序的结构，特别是累加器的概念，这个内容在后面会多次用到，必须充分理解。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、思考与探索参考答案

1. 思考与探索 1

累加器的初值为什么要设为 0？若累加器的初始值设为其他值会有什么影响？

解析：

0 加上任何值，结果还是这个值，不会影响结果。因此累加器初始值设为 0 不会影响累加的结果。如果设为其他值，不论正负，结果都会多出或少出这个初始值。

2. 思考与探索 2

累乘器的初值为什么要设为 1？若累乘器的初值设为 0 或其他值会有什么影响?

解析：

1 乘以任何值，结果还是这个值，不会影响结果。因此累乘器初值设为 1 不会影响累乘的结果。如果累乘器初值设为其他值，不论正负，结果都会多出或少出这个值的倍数。

五、练习提升参考答案

第 1 题：编程求 t=30×29×28×…×25 的值。

解析：

JavaScript 程序：

```
<SCRIPT>
t=1;
for (i=30;i>24;i--)
   t=t*i;
document.write(t);
</SCRIPT>
```

Python 程序：

```
t=1;
for i in range(30,24,-1):
    t=t*i;
print(t);
```

第 2 题：图 4-2-5 是输出斐波那契数列前 20 项的流程图。完成下列各题，并将流程图补充完整。（斐波那契数列是指：数列第 1 项和第 2 项均为 1，从第 3 项起以后各项均是它的前两项之和）

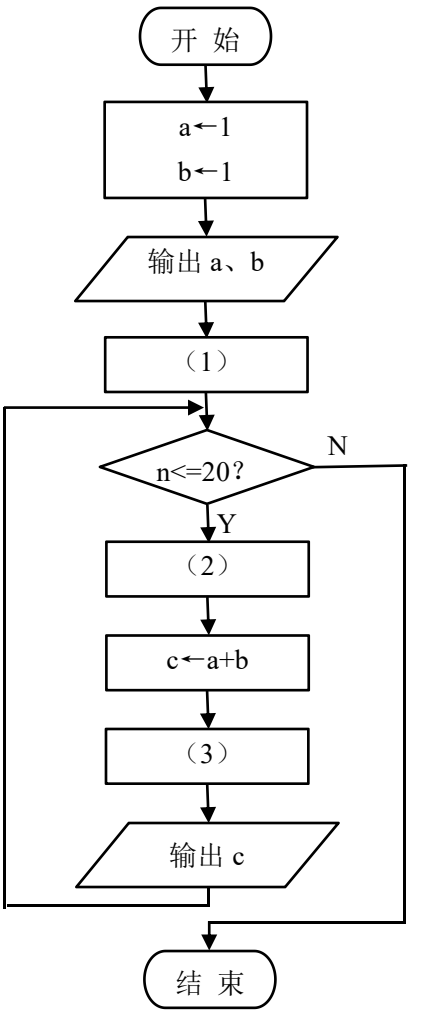

图 4-2-5　求斐波那契数列前 20 项流程图

（1）处的内容是____。

A. n←1　　B. n←2　　C. n←3　　D. n←4

（2）处的内容是____。

A. n←n+1　　B. n←n+2　　C. n←n+3　　D. n←n+4

（3）处的内容是____。

A. b←a　c←b　　B. a←b　c←b　　C. b←a　b←c　　D. a←b　b←c

解析：

（1）的答案是 C，（2）的答案是 A，（3）答案是 D

程序如下。

JavaScript 程序：

<SCRIPT>

```
a=1;b=1;
c=0;
document.write(a+"<br>");
document.write(b+"<br>");
for (i=2;i<20;i++)
{
  c=a+b;
  a=b;b=c;
  document.write(C+"<br>");
}
</SCRIPT>
```

Python 程序：

```
a=1;b=1;
c=0;
print(a);
print(b);
for n in range(2,20):
    c=a+b;
    a=b;b=c;
    print(c);
```

第三节　破 解 密 码

——循环的嵌套

一、教材分析

（一）内容分析

分支结构可以嵌套，循环结构也可以嵌套。本节讲述双重循环嵌套和多重循环嵌套，

通过破解密码和鸡兔同笼两个问题的解决，让学生体验循环的魅力和作用。本节内容对学生来说有一定难度。

（二）教学目标

（1）通过实践活动，掌握循环嵌套的执行过程。

（2）通过实践活动，学会利用循环语句和穷举算法的设计思想解决问题。

（三）教学重点和难点

1. 教学重点

（1）循环嵌套的执行过程。

（2）利用循环语句和穷举算法的设计思想解决问题。

2. 教学难点

（1）循环嵌套的执行过程。

（2）利用循环语句和穷举算法的设计思想解决问题。

二、学情分析

学生在本节学习之前，已具有顺序、分支、循环程序设计的基本思想，但对混合使用三种结构灵活解决问题并没有经验。本节的问题实现有一定难度，学生可能会有接受困难，需要深入思考才可以理解。本节内容是学生程序设计水平的分界点，有些学生可以顺利地理解并掌握，有些学生理解起来有一定难度，不要强求全部学生完全理解并掌握。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过模拟黑客技术——密码破解问题，让学生体验利用计算机的速度实现一定的目的，从而提高学生对程序代码的编写兴趣。

2. 实践活动，掌握技巧

通过“实践活动”使学生了解穷举法，这一方法是程序设计中的独特算法策略，这个内容在后面会多次用到，必须充分理解。通过对比“鸡兔同笼问题”的两种穷举方法，让学生理解，不同的算法设计思路会大大影响程序的执行效率。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、练习提升参考答案

编写程序，求 1!+2!+3!+….+10!的值。

解析：

JavaScript 程序：

```
<SCRIPT>
   S=0;
   for (i=1;i<=10;i++)
   {    T=1;
        for (j=1;j<=i;j++)
             T=T*j;
        S=S+T;
   }
   alert(S);
</SCRIPT>
```

Python 程序：

```
S=0;
for i in range(1,11):
      T=1;
      for j in range(1,i+1):
            T=T*j;
      S=S+T;
print(S);
```

第四节　程 序 推 理
——穷举法和逻辑判断应用

一、教材分析

（一）内容分析

本节内容属于扩展内容，难度比较大，也比较有趣味性，需要教师课前做好充分的教学准备，深刻理解教学案例中的程序设计思想，同时也需要再次理解数学建模在程序设计中的重要性。不同的数学模型，可以有不同的穷举方案，这是程序设计的魅力所在，也能初步体会到算法艺术的称呼并非言过其实。

（二）教学目标

（1）通过两个“实践活动”，掌握穷举算法的使用。

（2）通过实践活动，理解复杂条件的表示和处理。

（三）教学重点和难点

1. 教学重点

（1）穷举算法的使用。

（2）复杂条件的表示和处理。

2. 教学难点

（1）穷举算法的使用。

（2）复杂条件的表示和处理。

二、学情分析

学生在本节学习之前，已经明白穷举算法的基本思想，但在解决实际问题时，数学模型和条件比较复杂，需要进一步深入学习才能理解，本节的问题实现有较大难度，学生接受可能会有困难，不要强求全部学生完全理解并掌握。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

本节通过两个“实践活动”中的“计算机断案问题”和“排班问题”，提升学生的学习兴趣。随着程序设计学习的深入，编写程序的难度越来越大，必须想办法让学生坚持下去。

2. 实践活动，掌握技巧

通过“实践活动”体验程序高超的逻辑判断能力，再通过流程图了解程序的结构，特别是数学建模的概念，尽管不是第一次提到这个概念，但本节是学生充分认识数学建模的具体应用。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、实践活动指导

实践活动 1

警察抓小偷：警察抓到了 A、B、C、D 四名偷窃嫌疑人，其中一人是小偷，审问中，A 说“我不是小偷”，B 说“C 是小偷”，C 说“小偷肯定是 D”，D 说“C 在冤枉人”，有三人说真话，一人说假话，到底谁是小偷呢？

解析：

本“实践活动”给出了两种程序，一种程序采用了分支嵌套，另一种程序没有采用分支嵌套。

从运行结果可以发现这个没有分支嵌套程序的运行结果与前面有分支嵌套程序的运行结果相同，能起到同样的判断效果。

哪一个程序执行步数更少呢？从效率和程序可读性两方面对比两种程序的优势与不足。

分支嵌套程序的优点：

程序执行步数更少，程序执行效率高。本问题中条件语句的条件跟问题要求比较一致。是推荐的编程方法。

没有分支嵌套程序的优点：

程序可读性强，但条件设置会比问题描述复杂，需要再处理。

五、练习提升参考答案

第 1 题：一块金属，三个人对它进行判断。

甲说：不是铁，也不是钢。

乙说：不是铁，而是锌。

丙说：不是锌，而是铁。

结果有一个人全说错了，一个人全说对了，一个人对一句错一句。请判断这块金属是什么？

解析：

JavaScript 程序：

```
<SCRIPT>
  for (i=0;i<=2;i++)
  {
        S1=0;S2=0;S3=0;
        if (i!=0) S1=S1+1;
        if (i!=1) S1=S1+1;
        if (i!=0) S2=S2+1;
        if (i==2) S2=S2+1;
        if (i!=2) S3=S3+1;
        if (i==0) S3=S3+1;
        if (S1!=S2&&S1!=S3&&S2!=S3)
            document.write("i="+i);
  }
</SCRIPT>
```

Python 程序：

```
for i in range(0,3):
    S1=0;S2=0;S3=0;
    if i!=0:S1=S1+1;
    if i!=1:S1=S1+1;
```

```
    if i!=0:S2=S2+1;
    if i==2:S2=S2+1;
    if i!=2:S3=S3+1;
    if i==0:S3=S3+1;
    if S1!=S2 and S1!=S3 and S2!=S3:
        print("i=",i);
```

第 2 题：甲、乙、丙、丁、戊五个人在运动会上分别获得一百米、二百米、跳高、跳远和铅球冠军，有 A、B、C、D 四个人猜测比赛结果。

A 说：乙获得铅球冠军，丁获得跳高冠军。

B 说：甲获得一百米冠军，戊获得跳远冠军。

C 说：丙获得跳远冠军，丁获得二百米冠军。

D 说：乙获得跳高冠军，戊获得铅球冠军。

上述每个人都说对一句，说错一句。求甲、乙、丙、丁、戊各获得哪项冠军。

解析：

JavaScript 程序：

```
<SCRIPT>
  for (A=1;A<=5;A++)
  for (B=1;B<=5;B++)
  for (C=1;C<=5;C++)
  for (D=1;D<=5;D++)
  for (E=1;E<=5;E++)
  {
      if (A!=B&&A!=C&&A!=D&&A!=E&&B!=C&&B!=D&&B!=E&&C!=
D&&C!=E&&D!=E)
      {
          S1=0;S2=0;S3=0;S4=0;
          if (B==5) S1=S1+1;
          if (D==3) S1=S1+1;
          if (A==1) S2=S2+1;
          if (E==4) S2=S2+1;
          if (C==4) S3=S3+1;
          if (D==2) S3=S3+1;
          if (B==3) S4=S4+1;
          if (E==5) S4=S4+1;
          if (S1==1&&S2==1&&S3==1&&S4==1&&A+B+C+D+E==15)
```

```
                document.write("A="+A+"B="+B+"C="+C+"D="+D+"E="+E+"<br>");
            }
    }
</SCRIPT>
```

Python 程序：

```
for A in range(1,6):
    for B in range(1,6):
        for C in range(1,6):
            for D in range(1,6):
                for E in range(1,6):
                    if (A!=B and A!=C and A!=D and A!=E and B!=C and B!=D and B!=E and
C!=D and C!=E and D!=E):
                        S1=0;S2=0;S3=0;S4=0;
                        if (B==5):S1=S1+1;
                        if (D==3):S1=S1+1;
                        if (A==1):S2=S2+1;
                        if (E==4):S2=S2+1;
                        if (C==4):S3=S3+1;
                        if (D==2):S3=S3+1;
                        if (B==3):S4=S4+1;
                        if (E==5):S4=S4+1;
                        if (S1==1 and S2==1 and S3==1 and S4==1):
                            print("A=",A,"B=",B,"C=",C,"D=",D,"E=",E);
```

第五节　程序处理文件
——文件输入输出

一、教材分析

（一）内容分析

本节文件输入输出内容相对独立，主要为大数据量问题的处理做准备，以充分体现出计算机的速度优势，也为以后提交答案类问题做基础。

（二）教学目标

（1）通过“实践活动”——“成绩计算”（输入输出的要求与前面所学的不同），掌握文件输入输出的方法。

（2）通过“实践活动”——“成绩计算”，理解输入、输出的作用。

（三）教学重点和难点

1. 教学重点

（1）文件输入输出的方法。

（2）输入输出的作用。

2. 教学难点

文件处理及应用。

二、学情分析

学生在本节学习之前，已经明白程序设计的基本思想，体验到了计算机的作用，但在实际问题当中，数据规模可能比较大，输入数据的来源也不能只是键盘输入，计算机可以以不同的方式与外界进行数据传输。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过加大输入数据的数据量，以前所采用的键盘输入出现了问题，从而引入文件输入数据的概念。

2. 实践活动，掌握技巧

通过“实践活动”——成绩处理，让学生掌握文件输入、输出的方法。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、思考与探索参考答案

分别找出上例程序中进行读文件、写文件的语句，通过互联网查阅资料说出它们的功能。

解析：

JavaScript 程序：

```
<SCRIPT>
   var s,x,n,fin,fou;
   fso=new ActiveXObject("Scripting.FileSystemObject");
   fout=fso.createtextfile("c:/1/math.out");
   for (i=1;i<=6;i++)
   {
   //下面语句将数据文件打开，每次循环依次打开不同成绩文件
fin = fso.OpenTextFile("c:/1/math"+i+".txt");
n = Number(fin.Readline());//从成绩文件中读出人数
        s=0;
for (i=1;i<=n;i++)
```

```
        {    x=fin.Readline();    //从成绩文件中读入一行
             s=s+Number(x);
        }
fou.Writeline(i+"班平均分："+s/n);//向文件输出数据
}
</SCRIPT>
```

Python 程序：

```
m=6;
fout = open("c:/1/math.out","w");
for i in range(1,m+1):
        st="c:/1/math"+str(i)+".txt";
        fin=open(st);
        n = eval(fin.readline());#读出人数
        s=0;
           for j in range(1,n+1):
                x=fin.readline();    #从成绩文件中读入一行
             s=s+eval(x);
        fin.close();#关闭输入文件
           st=str(i)+"班平均分："+str(s/n)+"\n\r";
           fout.write(st);#向文件输出数据
fout.close();#关闭输出文件
```

五、练习提升参考答案

加法问题：有一些加法运算需要计算，这些运算问题的数据存储在一些文本文件（输入文件）中。现在需要编写程序进行计算，并将计算结果存放在答案文件（输出文件）中。

输入文件格式：

第 1 行包括一个正整数 *n*；

接下来有 2×*n* 行，每行只有一个数，连续的两行是一道需要计算的加法题的两个加数。

输出文件格式：

输出文件包括 *n* 行，每行一个整数，是第 *n* 道加法题计算的结果，也就是答案。

输入、输出样例：

plus.in

2

4

5
120
7
plus.ans
9
127

解析：

JavaScript 程序：

```
<SCRIPT>
   var fso=new ActiveXObject("Scripting.FileSystemObject");
   fin=fso.OpenTextFile("d:/1/a.txt");
   fout=fso.createtextfile("d:/1/b.out");
   x=fin.Readline();
   n=Number(x);
   for (var i=1;i<=n;i++)
   {
        x=fin.Readline();
        y=fin.Readline();
        s=Number(x)+Number(y);
        //document.write(s+"<BR>");
        fout.Writeline(s);
   }
   fin.close();fout.close();
</SCRIPT>
```

Python 程序：

```
fin=open("d:/1/a.txt");
fout=open("d:/1/b.out","w");
n=eval(fin.readline());
for i in range(1,n+1):
          x=fin.readline();
          y=fin.readline();
          s=eval(x)+eval(y);
          fout.write(str(s)+"\n");
fin.close();
```

fout.close();

六、教师知识延伸

Javascript 技术参考（六）

（一）操作文件（FSO）

FileSystemObject（FSO），对象模式，允许对大量的属性、方法和事件，使用较熟悉的 object.method 语法来处理文件夹和文件。

使用这个基于对象的工具和：

（1）HTML 来创建 Web 页；

（2）Windows Scripting Host 来为 Microsoft Windows 创建批文件；

（3）Script Control 来对用其他语言开发的应用程序提供编辑脚本的能力。

假定本文档使用 FSO 对象模式来创建由服务器端的 Internet Web 页执行的脚本，在客户端使用 FSO 可引起严重的安全性问题，其提供了潜在的不受欢迎的对客户端本地文件系统的访问，因为使用了服务器端，Internet Explorer 默认安全设置不允许客户端使用 FileSystemObject 对象覆盖那些默认值可能会引起在本地计算机上不受欢迎的对其文件系统的访问，从而导致文件系统完整性的破坏，同时引起数据遗失或更糟的情况。

FSO 对象模式使服务器端的应用程序能创建、改变、移动和删除文件夹，或探测特定的文件夹是否存在，若存在，还可以找出有关文件夹的信息，如名称、被创建或最后一次修改的日期等。

FSO 对象模式还使文件处理变得很容易。在处理文件时，主要的目标是以易于访问的格式把数据存储在有效的空间和资源中，这就要求能够创建文件，插入和改变数据，以及输出（读取）数据。因为把数据存储在数据库中，如 Access 或 SQL 服务器，会给应用程序增加很大的开销，所以把数据存储在二进制或文本文件中可能是最有效的解决方案。用户可能不希望有该开销，或者数据访问要求可能不需要与功能完备的数据库相关联的所有额外功能。

（二）处理文件

有两种主要的文件处理类型：

（1）创建文件、添加或删除数据、以及读取文件；

（2）移动、复制和删除文件。

1. 创建文件

创建空文本文件（有时又叫作“文本流”）有三种方法。

第一种方法是用 CreateTextFile 方法。下面的示例示范了在 VBScript 中如何用这种方

法来创建文本文件。

```
Dim fso,f1;
Set fso=CreateObject("Scripting.FileSystemObject");
Set f1=fso.CreateTextFile("c:/testfile.txt",True);
```

要在 JavaScript 中用这种方法，则使用下面的代码：

```
var fso,f1;
fso=new ActiveXObject("Scripting.FileSystemObject");
f1=fso.CreateTextFile("c://testfile.txt", true);
```

创建文本文件的第二种方法是使用 FileSystemObject 对象的 OpenTextFile 方法，并设置 ForWriting 标志。在 VBScript 中，代码如下：

```
Dim fso,ts;
Const ForWriting=2;
Set fso=CreateObject("Scripting.FileSystemObject");
Set ts=fso.OpenTextFile("c:/test.txt",ForWriting,True);
```

要在 JavaScript 中使用这种方法来创建文本文件，则使用下面的代码：

```
var fso,ts;
var ForWriting=2;
fso=new ActiveXObject("Scripting.FileSystemObject");
ts=fso.OpenTextFile("c:/test.txt",ForWriting,true);
```

创建文本文件的第三种方法是使用 OpenAsTextStream 方法，并设置 ForWriting 标志。要使用这种方法，在 VBScript 中使用下面的代码：

```
Dim fso,f1,ts;
Const ForWriting=2;
Set fso=CreateObject("Scripting.FileSystemObject");
fso.CreateTextFile("c:/test1.txt");
Set f1=fso.GetFile("c:/test1.txt");
Set ts=f1.OpenAsTextStream(ForWriting,True);
```

在 JavaScript 中，则使用下面示例中的代码：

```
var fso, f1, ts;
var ForWriting=2;
fso=new ActiveXObject("Scripting.FileSystemObject");
fso.CreateTextFile("c://test1.txt");
f1=fso.GetFile("c://test1.txt");
ts=f1.OpenAsTextStream(ForWriting, true);
```

2. 数据添加到文件

创建了文本文件后，可使用下面的三个步骤向文件添加数据：

打开文本文件→写入数据→关闭文件

打开现有的文件，可使用 FileSystemObject 对象的 OpenTextFile 方法或 File 对象的 OpenAsTextStream 方法。

使用 TextStream 对象的 Write、WriteLine 或 WriteBlankLines 方法，可写数据到打开的文本文件，如表 4.5.1 所示。

表 4.5.1　写数据到文件

方　　法	任　　务
Write	向打开的文本文件写数据，不用后续一个新行字符
WriteLine	向打开的文本文件写数据，后续一个新行字符
WriteBlankLines	向打开的文本文件写一个或多个空白行

使用 TextStream 对象的 Close 方法，可关闭一个打开的文件。

注意：新行字符包含一个或几个字符（取决于操作系统），以把光标移动到下一行的开始位置（回车/换行）。某些字符串末尾可能已经有这个非打印字符了。

下面的示例示范了在 VBScript 中打开文件和同时使用三种写入方法向文件添加数据，然后关闭文件。

```
Sub CreateFile();
Dim fso,tf;
   Set fso=CreateObject("Scripting.FileSystemObject");
   Set tf=fso.CreateTextFile("c:/testfile.txt", True);
tf.WriteLine("Testing 1, 2, 3.");   '写一行，并且带有新行字符
tf.WriteBlankLines(3);   '向文件写三个新行字符
tf.Write ("This is a test.");   '写一行
tf.Close;
End Sub
```

下面的示例示范了在 JavaScript 中如何使用这三种方法。

```
function CreateFile();
{
var fso,tf;
fso=new ActiveXObject("Scripting.FileSystemObject");
tf=fso.CreateTextFile("c://testfile.txt", true);
tf.WriteLine("Testing 1, 2, 3.");   //写一行，并且带有新行字符
tf.WriteBlankLines(3);   //向文件写三个新行字符
```

```
tf.Write ("This is a test.");   //写一行
tf.Close();
}
```

3. 读取文件

使用 TextStream 对象的 Read、ReadLine 或 ReadAll 方法，可从文本文件读取数据。如表 4.5.2 所示。

表 4.5.2　读取文件

方　　法	任　　务
Read	从文件读取指定数量的字符
ReadLine	读取一整行（一直到但不包括新行字符）
ReadAll	读取文本文件的整个内容

在使用 Read 或 ReadLine 方法时，如果想跳过数据的特殊部分，可使用 Skip 或 SkipLine 方法。Read 方法的结果文本存在一个字符串中，该字符串可以显示在一个控件中，也可以用字符串函数（如 Left、Right 或 Mid）来分析或连接。

下面的示例示范了在 VBScript 中打开文件和写数据到文件中，并从文件读取数据。

```
Sub ReadFiles;
Dim fso,f1,ts,s;
Const ForReading=1;
Set fso=CreateObject("Scripting.FileSystemObject");
Set f1=fso.CreateTextFile("c:/testfile.txt",True);
'写一行
Response.Write "Writing file <br>";
f1.WriteLine "Hello World";
f1.WriteBlankLines(1);
f1.Close;
'读取文件的内容
Response.Write "Reading file <br>";
Set ts=fso.OpenTextFile("c:/testfile.txt",ForReading);
s=ts.ReadLine;
Response.Write "File contents = '" & s & "'";
ts.Close;
End Sub
```

下面的示例示范了在 JavaScript 中打开文件和写数据到文件中，并从文件读取数据。

```
function ReadFiles();
```

```
{
var fso,f1,ts,s;
var ForReading=1;
fso=new ActiveXObject("Scripting.FileSystemObject");
f1=fso.CreateTextFile("c://testfile.txt", true);
//写一行
Response.Write("Writing file <br>");
f1.WriteLine("Hello World");
f1.WriteBlankLines(1);
f1.Close();
//读取文件的内容
Response.Write("Reading file <br>");
ts=fso.OpenTextFile("c://testfile.txt", ForReading);
s=ts.ReadLine();
Response.Write("File contents = '" + s + "'");
ts.Close();
}
```

Python 技术参考（六）

Python 有几个用于创建、读取、更新和删除文件的函数。

（一）创建文件

在 Python 中创建新文件，可使用 open()函数，并使用以下参数之一。

"x"：创建一个文件，如果文件存在则返回错误。

"a"：如果指定的文件不存在，则创建一个新文件。

"w"：如果指定的文件不存在，将创建一个文件。

下面的示例创建名为"myfile.txt"的文件。

```
f=open("myfile.txt","x");
```

结果：已创建新的空文件。

下面的示例，如果打开的文件不存在，则创建新文件。

```
f = open("myfile.txt","w");
```

（二）打开文件

在 Python 中，使用已有文件要先将文件打开，可使用 open()函数。

open()函数有两个参数：文件名和模式。

有四种打开文件的不同方法（模式）。

"r"：读取，是默认值，打开文件进行读取，如果文件不存在则报错。

"a"：追加，打开供追加的文件，如果不存在则创建该文件。

"w"：写入，打开文件进行写入，如果文件不存在则创建该文件。

"x"：创建，创建指定的文件，如果文件存在则返回错误。

此外，还可以指定文件作为二进制还是文本模式进行处理。

"t"：文本模式，默认值。

"b"：二进制模式（如图像）。

下面的示例示范了在 Python 中打开指定文件。

```
f=open("demofile.txt");
```

以上代码等同于：

```
f=open("demofile.txt","rt");
```

因为"r"（读取）和"t"（文本）是默认值，所以不需要指定它们。

注释：使用时要确保文件存在，否则可能收到错误信息。

假设有文件 demofile.txt，位于与 Python 相同的文件夹中，文件内容如下。

```
Hello! Welcome to demofile.txt
This file is for testing purposes.
Good Luck!
```

如需打开文件，可使用内建的 open()函数。

open()函数返回文件对象，此对象有一个 read()方法用于读取文件的内容，如下例。

```
f=open("demofile.txt","r");
print(f.read());
```

（三）读取文件

1. 只读取文件的一部分

默认情况下，read()方法返回整个文本，也可以指定要返回的字符数。

下面的示例在 Python 中返回文件中的前五个字符。

```
f = open("demofile.txt","r");
print(f.read(5));
```

2. 读行

可以使用 readline()方法返回一行，如：

```
f=open("demofile.txt","r");
print(f.readline());
```

通过两次调用 readline()方法，可以读取文件的前两行，如：

```
f=open("demofile.txt","r");
print(f.readline());
print(f.readline());
```

通过循环可遍历文件中的行，还可以逐行读取整个文件，如：

```
f=open("demofile.txt","r");
for x in f:
   print(x);
```

（四）写入文件

如需写入已有的文件，必须向 open()函数添加参数：

（1）"a"：追加，追加到文件的末尾；

（2）"w"：写入，覆盖任何已有的内容。

下面示例，打开文件“demofile2.txt”，并将内容追加到文件中。

```
f=open("demofile2.txt","a");
f.write("Now the file has more content!");
f.close();
#追加后，打开并读取该文件
f=open("demofile2.txt","r");
print(f.read());
```

打开文件“demofile3.txt”并覆盖内容，如：

```
f=open("demofile3.txt","w");
f.write("Woops! I have deleted the content!");
f.close();
#写入后，打开并读取该文件
f=open("demofile3.txt","r");
print(f.read());
```

注释："w"方法会覆盖全部内容。

（五）关闭文件

完成对文件的操作后，关闭文件是一个好习惯。

完成后关闭文件，如：

```
f=open("demofile.txt","r");
print(f.readline());
```

```
f.close();
```

注释：在某些情况下，由于缓冲，应该始终关闭文件，在关闭文件之前，对文件所做的更改可能不会显示。

（六）删除文件

如需删除文件，必须导入 OS 模块，并运行其 os.remove()函数。

删除文件“demofile.txt”，如：

```
import os;
os.remove("demofile.txt");
```

为避免出现错误，需要在删除文件之前检查该文件是否存在。

检查文件是否存在，若存在，将其删除：

```
import os;
if os.path.exists("demofile.txt"):
   os.remove("demofile.txt");
else:
   print("The file does not exist");
```

如需删除整个文件夹，可使用 os.rmdir()方法。

删除文件夹“myfolder”，如：

```
import os;
os.rmdir("myfolder");
```

提示：只能删除空文件夹。

第五章　批量数据处理

本章的主要内容是批量数据的处理，是一维数组的使用。结合前面讲的顺序程序设计、分支程序设计，特别是利用程序的循环语句与数组相配合，解决实际生活中的相关问题。本章共四节，前两节介绍数据和排序，后两节介绍字符串处理。

第一节　变量也能批量用
——数组

一、教材分析

（一）内容分析

本节重点讲解数组的概念和应用，通常与循环程序配合使用。通过让学生按步骤实施“实践活动”，了解通过循环程序处理数组的方法。

（二）教学目标

（1）通过实践活动和总结讲解，掌握通过循环语句操作数组的方法。
（2）通过实践活动和总结讲解，理解数组的概念和应用方法。

（三）教学重点和难点

1. 教学重点

（1）数组的使用方法。

（2）数组的概念及应用。

2. 教学难点

数组的使用方法。

二、学情分析

学生在本节学习之前，对顺序结构、分支结构、循环结构的程序有一定了解，能编写一些程序，但对于批量数据如何处理，还需要通过本节进一步学习。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过加大处理数据的数据量，以前所采用的单个变量处理数据出现了困难，从而引入数组的概念。

2. 实践活动，掌握技巧

通过“实践活动”中的“批量数据求最值”，让学生掌握数组的使用方法。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、练习提升参考答案

第 1 题：编写程序：从键盘输入 10 个数，按输入顺序的倒序输出。

解析：

JavaScript 程序：

<SCRIPT>

```
  var a=[];
  n=10;
  for (i=1;i<=n;i++)     //读入 10 个数
       a[i]=Number(prompt("输入：  "));
  for (i=n;i>=1;i--)     //输出 10 个数
       document.write(a[i]+" ");
</SCRIPT>
```

Python 程序：

```
a=[0]*11;
n=10;
for i in range(1,n+1):   #读入 10 个数
     a[i]=eval(input("输入第"+str(i)+"个数：  "));
for i in range(n,0,-1):              #逆序输出
     print(a[i]);
```

第 2 题：编写程序：从键盘输入 10 个数，求前 5 个数中的最大值，后 5 个数中的最小值。

解析：

JavaScript 程序：

```
<SCRIPT>
  var a=new Array(11);
  n=10;
  for (i=1;i<=n;i++)   //读入 10 个数
       a[i]=Number(prompt("输入:"));
  max=a[1];   //求最大值
  for (i=2;i<=5;i++)
       if (a[i]>max) max=a[i];
  document.write("前 5 个数中的最大数为：  "+max+"<BR>");
 min=a[6];
 for (i=7;i<=n;i++)
       if (a[i]<min) min=a[i];
  document.write("后 5 个数中的最小数为：  "+min+"<BR>");
  for (i=n;i>=1;i--)
       document.write(a[i]+"<BR>");
</SCRIPT>
```

Python 程序：

```
a=[0 for i in range(1,12)];
n=10;
for i in range(1,n+1):   #读入 10 个数
    a[i]=eval(input("输入第"+str(i)+"个数"));
max=a[1];                    #max 初始化为第一个数
for i in range(2,6):         #max 和第 2－5 个依次比较
    if a[i]>max:
        max=a[i];
print("前 5 个数中的最大数是",max);
min=a[6];
#max 初始化为第一个数
for i in range(7,n+1):          #max 和第 2－5 个依次比较
    if a[i]<min:
        min=a[i];
print("后 5 个数中的最小数是",min);
for i in range(n,0,-1):
    print(a[i]);
```

五、教师知识延伸

JavaScript 技术参考（七）

JavaScript 数组

JavaScript 数组是一种特殊的变量，用于在单一变量中存储多个值。如：

```
var cars=["比亚迪","长城","红旗"];
```

有一个项目清单（例如，汽车品牌列表），可以在单个变量中存储汽车品牌，如：

```
var car1="比亚迪";
var car2="长城";
var car3="红旗";
```

不过，假如希望遍历所有汽车并找到一个特定的值？假如不是三个汽车品牌而是三百个呢？

解决方法就是数组！

数组可以用一个单一的名称存放很多值，并且还可以通过引用索引号来访问这些值。

1. 创建数组

使用数组文本是创建 JavaScript 数组最简单的方法，如：

```
var array-name=[item1,item2,…];
```

【例 1】

```
var cars=["比亚迪","长城","红旗"];
```

空格和折行并不影响。声明可横跨多行，如：

```
var cars=[
    "比亚迪",
    "长城",
    "红旗"
];
```

最后一个元素之后不能写逗号（如"红旗",），否则可能存在跨浏览器兼容性问题。

使用 JavaScript 关键词 new 也可以创建数组，并为其赋值，如：

```
var cars=new Array("比亚迪","长城","红旗");
```

以上两个例子效果完全一样。出于简洁、可读性和执行速度的考虑，最好使用第一种方法（数组文本方法）。

2. 访问数组元素

通过引用索引号（下标号）来引用某个数组元素。

下面语句访问 cars 中的首个元素的值：

```
var name=cars[0];
```

【例 2】

```
var cars=["比亚迪","长城","红旗"];
document.getElementById("demo").innerHTML=cars[0];
```

[0]是数组中的第一个元素，[1]是第二个。数组索引从 0 开始。

3. 改变数组元素

下面语句修改了 cars 中第一个元素的值：

```
cars[0]="长安";
```

【例 3】

```
var cars=["比亚迪","长城","红旗"];
cars[0]="长安";
document.getElementById("demo").innerHTML=cars[0];
```

3. 访问完整数组

在 JavaScript 中，可通过引用数组名来访问完整数组，如：

```
var cars=["比亚迪","长城","红旗"];
document.getElementById("demo").innerHTML=cars;
```

4. 数组是对象

数组是一种特殊类型的对象。在 JavaScript 中，对数组使用 typeof 运算符会返回"object"。但是，JavaScript 数组最好以数组来描述。

数组使用数字来访问其元素。下例中，person[0]返回 Bill：

```
var person=["Bill","Gates",62];
```

对象使用名称来访问其“成员”。下例中，person.firstName 返回 Bill：

```
var person={firstName:"John",lastName:"Doe",age:46};
```

5. 数组元素可以是对象

JavaScript 变量可以是对象，数组是特殊类型的对象。

正因如此，可以在相同数组中存放不同类型的变量。

可以在数组中保存对象，还可以在数组中保存函数，甚至可以在数组中保存数组，如：

```
myArray[0]=Date.now;
myArray[1]=myFunction;
myArray[2]=myCars;
```

6. 数组的属性和方法

JavaScript 数组的真实力量隐藏在数组的属性和方法中，如：

```
var x=cars.length;     // length 属性返回元素的数量
var y = cars.sort();     // sort()方法对数组进行排序
```

数组的 length 属性返回数组的长度（数组元素的数目），如：

```
var fruits=["香蕉","苹果","橘子","芒果"];
fruits.length;                                  // fruits 的长度是 4
```

length 属性始终大于数组最后元素的索引（下标）。

7. 访问第一个数组元素

访问第一个数组元素，如：

```
fruits=["香蕉","苹果","橘子","芒果"];
var first=fruits[0];
```

8. 访问最后一个数组元素

访问最后一个数组元素，如：

```
fruits=["香蕉","苹果","橘子","芒果"];
```

```
var last=fruits[fruits.length-1];
```

9. 遍历数组元素

遍历数组的最安全方法是使用 for 循环，如：

```
var fruits,text,fLen,i;
fruits=["香蕉","苹果","橘子","芒果"];
fLen=fruits.length;
text="<ul>";
for (i=0;i<fLen;i++) {
        text+="<li>"+fruits[i]+"</li>";
}
```

也可以使用 Array.foreach()函数，如：

```
var fruits, text;
fruits=["香蕉","苹果","橘子","芒果"];
text="<ul>";
fruits.forEach(myFunction);
text+="</ul>";
function myFunction(value) {
   text+="<li>"+value+"</li>";
}
```

10. 添加数组元素

向数组添加新元素的最佳方法是使用 push()方法，如：

```
var fruits=["香蕉","苹果","橘子","芒果"];
fruits.push("梨");                          // 向 fruits 添加一个新元素(梨)
```

也可以使用 length 属性向数组添加新元素，如：

```
var fruits=["香蕉","苹果","橘子","芒果"];
fruits[fruits.length]="梨";        // 向 fruits 添加一个新元素(梨)
```

添加最高索引的元素可在数组中创建未定义的“洞”，如：

```
var fruits=["香蕉","苹果","橘子","芒果"];
fruits[6]="柠檬";                          // 向 fruits 添加一个新元素（柠檬）
```

Python 技术参考（七）

Python 数组

Python 没有内置对数组的支持，但可以使用 Python 列表代替。

数组用于在单个变量中存储多个值。

创建一个包含汽车品牌的数组，如：

```
cars=["比亚迪","长城","红旗"];
```

有一个项目清单（例如，汽车品牌列表），可以在单个变量中存储汽车品牌，如：

```
car1="比亚迪";
car2="长城";
car3="红旗";
```

不过，假如您希望遍历所有汽车并找到一个特定的值？假如不是三个汽车品牌而是三百个呢？

解决方案是数组！

数组可以用一个单一的名称存放很多值，并且还可以通过引用索引号来访问这些值。

1. 访问数组元素

通过索引号可以引用数组元素。

获取首个数组项目的值，如：

```
x=cars[0];
```

修改首个数组项目的值，如：

```
cars[0]="长安";
```

2. 数组长度

使用 len()方法可以返回数组的长度（数组中的元素数量）。

返回 cars 数组中的元素数量，如：

```
x=len(cars);
```

注释：数组长度总是比最高的数组索引大一个。

3. 循环数组元素

使用 for in 循环可以遍历数组的所有元素。

输出 cars 数组中的每个项目，如：

```
for x in cars:
   print(x);
```

4. 添加数组元素

使用 append()方法可以把元素添加到数组中。

向 cars 数组再添加一个元素：

```
cars.append("奇瑞");
```

5. 删除数组元素

使用 pop()方法可以从数组中删除元素。

删除 cars 数组的第二个元素：

```
cars.pop(1);
```

使用 remove()方法也可以从数组中删除元素。

删除值为“长城”的元素：

cars.remove("长城");

Python 列表的方法如表 5.5.1 所示。

表 5.5.1　Python 列表的方法

方　法	功　　能
append()	在列表的末尾添加一个元素
clear()	删除列表中的所有元素
copy()	返回列表的副本
count()	返回具有指定值的元素数量。
extend()	将列表元素（或任何可迭代的元素）添加到当前列表的末尾
index()	返回具有指定值的第一个元素的索引
insert()	在指定位置添加元素
pop()	删除指定位置的元素
remove()	删除具有指定值的项目
reverse()	颠倒列表的顺序

说明：列表的 remove()方法仅删除首次出现的指定值。

6. 数组方法

Python 提供一组可以在列表或数组上使用的内建方法。

说明：Python 没有内置对数组的的支持，但可以使用 Python 列表代替。

第二节　排 定 次 序
——排　序

一、教材分析

（一）内容分析

本节通过实践活动，让学生了解经典排序算法，学会利用双重循环实现选择排序的算法。实际上排序才是学习中遇到的第一个真正意义上的算法，学生会对算法思想有更为深刻的认识。

（二）教学目标

（1）通过实践活动，掌握选择排序算法的实现方法。

（2）通过实践活动，理解排序策略。

（三）教学重点和难点

1. 教学重点

（1）选择排序算法的实现方法。

（2）排序策略。

2. 教学难点

排序策略。

二、学情分析

本节是学生进一步理解算法思想的重要阶段，主要任务是通过实践活动理解选择排序算法，探索其他排序算法，并掌握其规律。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过介绍数值计算与非数值计算，引入排序的概念。

2. 实践活动，掌握技巧

通过“实践活动”——按顺序输出数据，从两个数据、三个数据到多个数据，使学生理解选择排序的基本原理，从而掌握选择排序算法。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、练习提升参考答案

编写程序：从键盘输入 10 个数，按从大到小排序并按从小到大的顺序输出。如果将 10 个数从小到大排序，程序应该如何修改？

解析：

JavaScript 程序：

```
<SCRIPT>
   var a=new Array(100);
   n=10;
   for (i=1;i<=n;i++)
      a[i]=Number(prompt("输入第"+i+"个数："));
   for (i=1;i<n;i++)
   for (j=i+1;j<=n;j++)
         if (a[j]>a[i])
        {
            temp=a[i];
            a[i]=a[j];
```

```
            a[j]=temp;
        }
   for (i=1;i<=n;i++)
        document.write(a[i]+" ");
</SCRIPT>
```

Python 程序：

```
a=[0 for i in range (1,100)];
n=10;
for i in range(1,n+1):
      a[i]=eval(input("输入第"+str(i)+"个数"));
for i in range(1,n):
     for j in range(i+1,n+1):
          if a[i]>a[j]:
                temp=a[i];
                a[i]=a[j];
                a[j]=temp;
for i in range(1,n+1): #输出结果
      print(a[i]," ");
```

第三节　数 据 压 缩
——字符串处理

一、教材分析

（一）内容分析

本节内容是一个综合练习，用到的知识点较多，也有新的内容——对字符串的理解。

（二）教学目标

（1）通过实践活动，让学生分析问题，掌握字符串的处理方法。

（2）通过实践活动，让学生分析问题，理解计算机处理字符串的方式。

（三）教学重点和难点

1. 教学重点

（1）字符串的处理方法。

（2）计算机处理字符串的方式。

2. 教学难点

计算机处理字符串的方式。

二、学情分析

学生在本节学习之前，已具有顺序、分支、循环程序设计的基本思想，但对三种结构混合使用，灵活解决问题并没有经验。本节问题的实现有一定难度，学生可能会有接受困难，需要深入思考才可以理解。本节内容是学生程序设计水平的分界点，有些学生可以顺利地理解并掌握，有些学生理解起来有一定难度，不要强求全部学生完全理解并掌握。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过提出数据压缩这个实际问题的应用，从解决问题的角度提高学生学习兴趣，从而引入字符串的概念和压缩算法的实现。

2. 实践活动，掌握技巧

通过“实践活动”——数据压缩问题，使学生压缩算法的实现方法。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、练习提升参考答案

第 1 题：有一个文本文件 in.txt，里面是一个二值图像的数据，只有“0”“1”两种符号，运用本节中学习的压缩算法对其进行压缩，将结果输出到 out.txt 中。

解析：

输入文件格式：包括一个字符串，表示二值图像的数据。

输出格式：若干行，每行一个整数和一个字符，中间用一个空格隔开，整数表示数量，字符为“0”或“1”。

JavaScript 程序：

```
<SCRIPT>
    var fso=new ActiveXObject("Scripting.FileSystemObject");
    fin=fso.OpenTextFile("d:/1/in.txt");
    fout=fso.createtextfile("d:/1/out.txt");
    st=fin.Readline();
    st=st+"#";
    document.write(st+"<BR>");
     j=1;
     for    (i=0;i<st.length-1;i++)
        if (st[i]==st[i+1])
           j=j+1;
       else
     {       document.write(j+" "+st[i]+"<BR>");
             fout.writeline(j+" "+st[i]);
             j=1;
      }
   fin.close();fout.close();
</SCRIPT>
```

Python 程序：

```
fin=open("d:/1/a.txt");
fout=open("d:/1/b-py.out","w");
st=fin.readline();
st=st+"#";
j=1;print(st);
for i in range(0,len(st)-1):
```

```
        print(str(i)+" "+st[i]);
        if (st[i]==st[i+1]):
            j=j+1;
        else:
            fout.write(str(j)+" "+st[i]+"\n");
            j=1;
    fin.close();
    fout.close();
```

容易出现的错误：如果用绝对路径"d:/1/a.txt"，不能误写为"d:\1\a.txt",否则浏览器可能执行出错。

第 2 题：现在有一个文本文件 sfz.txt，包含 100 个人的身份证号，编写程序统计这 100 个人中未成年人的数量。

输入文件格式：包括 100 行，每行有一个长度为 18 的字符串，表示一个人的身份证号。

输出格式：一个整数，表示其中未成年人的数量。

根据图 5-3-3 所示的程序流程图编写程序。

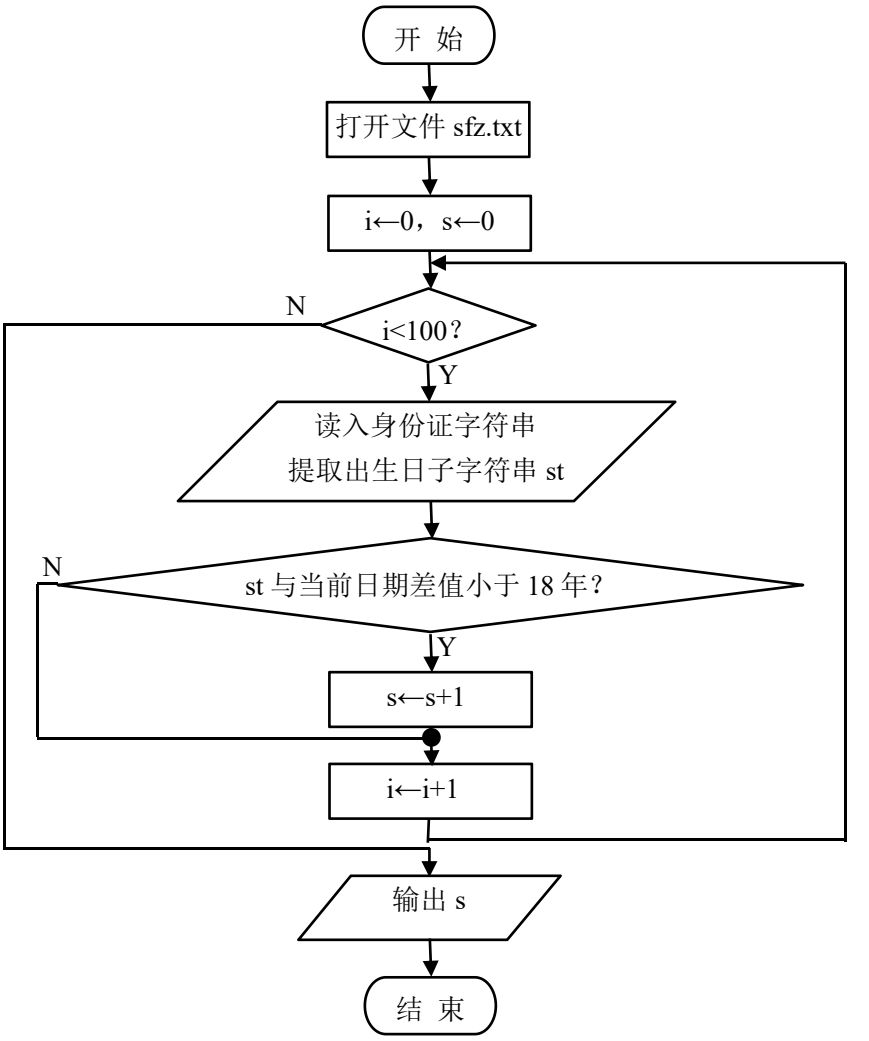

图 5-3-3　输出未成年人数量流程图

解析：

JavaScript 程序：

```
<SCRIPT>
   var fso=new ActiveXObject("Scripting.FileSystemObject");
   fin=fso.OpenTextFile("d:/1/sfz.txt");
   fout=fso.createtextfile("d:/1/sfz.out");
   var b=new Date(); //获取当前时间
   x1=eval(b.getFullYear())*10000;
   y1=(eval(b.getMonth())+1)*100;
   z1=eval(b.getDate());
   s1=0+x1+y1+z1;
```

```
   n=4;s=0;
   for (i=1;i<=n;i++){
        st=fin.Readline();
        x2=Number(st.substring(6,10))*10000;
        y2=Number(st.substring(10,12))*100;
        z2=Number(st.substring(12,14));
        s2=0+x2+y2+z2;
        if (s1-s2<180000) s++;
}
   fout.Writeline(s);
   fin.close();fout.close();
</SCRIPT>
```

Python 程序：

```
import datetime;
fin=open("d:/1/sfz.txt");
fout=open("d:/1/sfz.out","w");
d=datetime.datetime.now();
x1=d.year*10000;
y1=d.month*100;
z1=d.day;
s1=0+x1+y1+z1;
n=4;s=0;
for i in range(1,n+1):
        x=fin.readline();
        x2=int(x[6:10])*10000;
        y2=int(x[10:12])*100;
        z2=int(x[12:14]);
        s2=0+x2+y2+z2;
        if (s1-s2<180000):
             s=s+1;
fout.write(str(s));
fin.close();fout.close();
```

五、教师知识延伸

JavaScript 技术参考（八）

JavaScript 字符串

JavaScript 字符串是引号中的零个或多个字符，用于存储和操作文本，如：

```
var x="Xiao Qiang";
```

描述字符串的引号可以是单引号或双引号。

```
var carname="比亚迪 宋";
var carname='比亚迪 宋';
```

可以在字符串中使用引号，只要不匹配描述字符串的引号即可，如：

```
var answer="It's good to see you again!";
var answer="He is called 'Xiao Qiang'";
var answer='He is called "Xiao Qiang"';
```

1. 字符串长度

内建属性 length 可返回字符串的长度。

```
var txt="ABCDEFGHIJKLMNOPQRSTUVWXYZ";
var sln=txt.length;
```

2. 特殊字符

由于字符串必须由引号描述，对于下面的语句，JavaScript 会产生误解。

```
var y="中国是瓷器的故乡，因此 china 与"China（中国）"同名。";
```

该字符串将被认为是"中国是瓷器的故乡，因此 china 与"。

避免此问题的解决方法是使用转义字符\。

反斜杠转义符把特殊字符转换为字符串字符，如表 5.3.1 所示。

表 5.3.1 反斜杠转义符

代 码	结 果	描 述
\'	'	单引号
\"	"	双引号
\\	\	反斜杠

序列\"在字符串中插入双引号，如：

```
var x="中国是瓷器的故乡，因此 china 与\"China（中国）\"同名。";
```

序列\'在字符串中插入单引号，如：

```
var x = 'It\'s good to see you again';
```

序列\\在字符串中插入反斜杠，如：

```
var x = "字符 \\ 被称为反斜杠。";
```

转义字符（\）也可用于在字符串中插入其他特殊字符。

其他六个 JavaScript 中有效的转义符，如表 5.3.2 所示。

表 5.3.2　其他转义符

代　码	结　果
\b	退格键
\f	换页
\n	新行
\r	回车
\t	水平制表符
\v	垂直制表符

这六个转义字符最初设计用于控制打字机、电传打字机和传真机。它们在 HTML 中没有任何意义。

3. 长代码行换行

为了最佳可读性，程序员们通常会避免每行代码超过 80 个字符串。

如果某个 JavaScript 语句不适合一整行，那么最佳换行位置是某个运算符之后，如：

```
document.getElementById("demo").innerHTML=
"Hello Kitty.";
```

也可以用反斜杠在字符串中换行，如：

```
document.getElementById("demo").innerHTML = "Hello \
Kitty!";
```

\方法并不是 JavaScript 的标准，某些浏览器也不允许\字符之后的空格。

对长字符串换行的最安全做法（但是有点慢）是使用字符串加法，如：

```
document.getElementById("demo").innerHTML="Hello"+
"Kitty!";
```

不能通过反斜杠对代码行进行换行：

```
document.getElementById("demo").innerHTML= \
"Hello Kitty!";
```

Python 技术参考（八）

Python 字符串

1. 字符串字面量

python 中的字符串字面量由单引号或双引号括起。'hello'等同于"hello"。

使用 print()函数可以显示字符串字面量，如：

```
print("Hello");
print('Hello');
```

2. 用字符串向变量赋值

通过变量名称后跟等号和字符串，可以把字符串赋值给变量，如：

```
a="Hello";
print(a);
```

3. 多行字符串

使用三个引号可以将多行字符串赋值给变量。

使用三个双引号，如：

```
a="""Python is a widely used general-purpose, high level programming language.
It was initially designed by Guido van Rossum in 1991
and developed by Python Software Foundation.
It was mainly developed for emphasis on code readability,
and its syntax allows programmers to express concepts in fewer lines of code.""";
print(a);
```

也可以使用三个单引号，如：

```
a='''Python is a widely used general-purpose, high level programming language.
It was initially designed by Guido van Rossum in 1991
and developed by Python Software Foundation.
It was mainly developed for emphasis on code readability,
and its syntax allows programmers to express concepts in fewer lines of code.''';
print(a);
```

说明：在结果中，换行符插入与代码中相同的位置。

4. 字符串是数组

Python 中的字符串是表示 unicode 字符的字节数组。

Python 没有字符数据类型，单个字符就是长度为 1 的字符串。

方括号可用于访问字符串的元素。

获取位置 1 处的字符（请记住第一个字符的位置为 0），如：

```
a="Hello, World!";
print(a[1]);
```

（1）裁切。使用裁切语法可以返回一定范围的字符。指定开始索引和结束索引，以冒号分隔，以返回字符串的一部分。

获取从位置 2 到位置 5（不包括）的字符：

```
b="Hello,World!";
print(b[2:5]);
```

（2）负的索引。使用负索引从字符串末尾开始切片。

获取从位置 5 到位置 1 的字符，从字符串末尾开始计数。

```
b="Hello,World!";
print(b[-5:-2]);
```

（3）字符串长度。使用 len()函数可获取字符串的长度。

```
a="Hello,World!";
print(len(a));
```

5. 字符串方法

Python 有一组可用于字符串的内置方法。

（1）strip()方法删除开头和结尾的空白字符。

```
a=" Hello, World! ";
print(a.strip()) # returns "Hello, World!";
```

（2）lower()返回小写的字符串。

```
a="Hello,World!";
print(a.lower());
```

（3）upper()方法返回大写的字符串。

```
a="Hello,World!";
print(a.upper());
```

（4）replace()用另一段字符串来替换字符串。

```
a="Hello,World!";
print(a.replace("World","Kitty"));
```

（5）split()方法在找到分隔符的实例时将字符串拆分为子字符串。

```
a="Hello,World!";
print(a.split(",")) # returns ['Hello','World!'];
```

6. 检查字符串

使用 in 或 not in 关键字可以检查字符串中是否存在特定短语或字符。

检查以下文本中是否存在短语 ina。

```
txt="China is a great country";
x="ina" in txt;
print(x);
```

检查以下文本中是否没有短语 ain。

```
txt="China is a great country";
x="ain" not in txt;
print(x);
```

7. 字符串级联（串联）

使用+运算符可以串联或组合两个字符串。将变量 a 与变量 b 合并到变量 c 中：

```
a="Hello";
b="World";
c=a+b;
print(c);
```

在它们之间添加一个空格。

```
a="Hello";
b="World";
c=a+" "+b;
print(c);
```

8. 字符串格式

不能像下面实例组合字符串和数字。

```
age=63;
txt="My name is Bill, I am "+age;
print(txt);
```

可以使用 format()方法组合字符串和数字。format()方法接受传递的参数，格式化它们，并将它们放在占位符{}所在的字符串中。使用 format()方法将数字插入字符串。

```
age=63;
txt="My name is Bill, and I am {}";
```

```
print(txt.format(age));
```

format()方法接受不限数量的参数，并放在各自的占位符中。

```
quantity=3;
itemno=567;
price=49.95;
myorder="I want {} pieces of item {} for {} dollars.";
print(myorder.format(quantity,itemno,price));
```

使用索引号{0}可以确保参数被放在正确的占位符中。

```
quantity=3;
itemno=567;
price=49.95;
myorder="I want to pay {2} dollars for {0} pieces of item {1}.";
print(myorder.format(quantity,itemno,price));
```

9. 字符串方法

Python 有一组可以在字符串上使用的内建方法，如表 5.3.3 所示。

表 5.3.3　字符串方法

方　　法	描　　述
capitalize()	把首字符转换为大写
casefold()	把字符串转换为小写
center()	返回居中的字符串
count()	返回指定值在字符串中出现的次数
encode()	返回字符串的编码版本
endswith()	如果字符串以指定值结尾，则返回 true
expandtabs()	设置字符串的 tab 尺寸
find()	在字符串中搜索指定的值并返回它被找到的位置
format()	格式化字符串中的指定值
format_map()	格式化字符串中的指定值
index()	在字符串中搜索指定的值并返回它被找到的位置
isalnum()	如果字符串中的所有字符都是字母数字，则返回 True

续表

方　　法	描　　述
isalpha()	如果字符串中的所有字符都在字母表中，则返回 True
isdecimal()	如果字符串中的所有字符都是小数，则返回 True
isdigit()	如果字符串中的所有字符都是数字，则返回 True
isidentifier()	如果字符串是标识符，则返回 True
islower()	如果字符串中的所有字符都是小写，则返回 True
isnumeric()	如果字符串中的所有字符都是数，则返回 True
isprintable()	如果字符串中的所有字符都是可输出的，则返回 True
isspace()	如果字符串中的所有字符都是空白字符，则返回 True
istitle()	如果字符串遵循标题规则，则返回 True
isupper()	如果字符串中的所有字符都是大写，则返回 True
join()	把可迭代对象的元素连接到字符串的末尾
ljust()	返回字符串的左对齐版本
lower()	把字符串转换为小写
lstrip()	返回字符串的左修剪版本
maketrans()	返回在转换中使用的转换表
partition()	返回元组，其中的字符串被分为三部分
replace()	返回字符串，其中指定的值被替换为指定的值
rfind()	在字符串中搜索指定的值，并返回它被找到的最后位置
rindex()	在字符串中搜索指定的值，并返回它被找到的最后位置
rjust()	返回字符串的右对齐版本
rpartition()	返回元组，其中字符串分为三部分
rsplit()	在指定的分隔符处拆分字符串，并返回列表
rstrip()	返回字符串的右边修剪版本

续表

方　法	描　述
split()	在指定的分隔符处拆分字符串，并返回列表
splitlines()	在换行符处拆分字符串并返回列表
startswith()	如果以指定值开头的字符串，则返回 true
strip()	返回字符串的剪裁版本
swapcase()	切换大小写，小写成为大写，反之亦然
title()	把每个单词的首字符转换为大写
translate()	返回被转换的字符串
upper()	把字符串转换为大写
zfill()	在字符串的开头填充指定数量的 0 值

说明：所有字符串方法都返回新值。它们不会更改原始字符串。

第四节　加密和解密

——字符串处理

一、教材分析

（一）内容分析

本节介绍一种简单的加密算法及与其对应的解密算法，实际的加密算法比本节的算法要复杂，但本节内容有助于以后理解实用的加密算法。本节内容难度不是很大，比较有趣味性，需要教师课前做好充分的教学准备，深刻理解案例中的程序设计的思想。

（二）教学目标

（1）通过“实践活动”——“加密”，理解加密的算法思想。

（2）通过“实践活动”——“加密”，掌握加密算法。

（3）通过“思考与探索”，理解解密的算法思想。

（三）教学重点和难点

1. 教学重点

（1）加密和解密的算法思想。

（2）加密算法和解密算法。

2. 教学难点

（1）加密算法的思想。

（2）加密算法的程序实现。

二、学情分析

学生在本节学习之前，已经具备程序设计的基本思想和方法，但对文件的加密原理不太了解。本节的问题实现难度不大，比较有趣，学生接受并不困难。

三、教学建议

（一）课前准备

设备：计算机教室，用 JavaScript 教学的学校，保证 Windows 操作系统及“记事本”“IE 浏览器”或其他浏览器可正常使用；用 Python 教学的学校，保证 Python 3.x IDE 软件可正常使用。

学生：预习本节教材。

教师：熟悉上述软件的使用方法，并准确完成“实践活动”中的内容。

（二）过程设计

1. 问题引入，激发兴趣

通过提出加密这个实际问题的应用，从解决问题的角度提高学生学习兴趣，从而引入加密算法的实现。

2. 实践活动，掌握技巧

通过“实践活动”——数据加密问题，使学生压缩算法的实现方法。

3. 总结要点，扩展知识

根据学生“实践活动”完成情况，进行要点总结。

四、练习提升参考答案

有些时候，需要加密的文本比较多，不是一两句，而是整篇文章，怎么处理？将本节程序进行修改，使其能对一个文本文件进行加密。

解析：

JavaScript 程序：

```
<SCRIPT>
   var fso=new ActiveXObject("Scripting.FileSystemObject");
   fin=fso.OpenTextFile("d:/1/mingwen.txt");
   fout=fso.createtextfile("d:/1/miwen.out");
   st=fin.Readline();
   for (i=0;i<st.length;i++)
   {
     if ((st[i]>='a'&&st[i]<='z')||(st[i]>='A'&&st[i]<='Z')) //是否为字母
        {
              if (st[i]=='z' || st[i]=='Z')
              {
                   if (st[i]=='z')
                         //document.write('a');
                         fout.write('a');
                   else
                         //document.write('A');
                         fout.write('A');
               }
              else //输出后一个字母
              {
                    c=st[i].charCodeAt();
                   // document.write(String.fromCharCode(c+1));
                   fout.write(String.fromCharCode(c+1));
               }
       }//if
      else   //非字母原样输出
           //document.write(st[i]);
            fout.write(st[i]);
   }//for
```

</SCRIPT>

Python 程序：

```
fin=open("d:/1/mingwen.txt");
fout=open("d:/1/miwen.out","w");
st=fin.readline();
for i in range(0,len(st)):
   if (st[i]>='a' and st[i]<='z')or(st[i]>='A' and st[i]<='Z'):
        if st[i]=='z' or st[i]=='Z':
            if st[i]=='z':
                #print('a',end="");
                fout.write('a');
            else:
                #print('A',end="");
                fout.write('A');
        else:
            #print(chr(ord(st[i])+1),end="");
            fout.write(chr(ord(st[i])+1));
   else:
        #print(st[i],end="");
        fout.write(st[i]);
fin.close();fout.close();
```

后　记

一、教学建议

（一）教学方法建议

尽管教材在提供教学范例时，选择了两种计算机语言的程序代码，但并不是要求师生将两种计算机语言都掌握，这也不符合大多数学生的学习习惯和接受能力。教师在进行教学的时候，可以选择两种计算机语言之一进行教学。建议不要把两种计算机语言混合教学。这样，学生学习起来容易混淆，因为两种计算机语言的关键字、语句格式等都有所不同。如果有个别能力较强的学生对两种计算机语言都比较感兴趣，可以个别处理，例如可以建议其在学习完第四章后再学习第二种计算机语言。前四章学习完成后，学生对计算机语言的基本语法，语句格式等都会相对熟悉，利用已掌握的内容也能较快掌握第二种计算机语言。

（二）对于本书两种编程语言的说明

教材采用 JavaScript 这样的脚本语言作为首选的教学语言之一，其原因是显而易见的，这种语言不依托于某种软件开发环境（IDE），也没有专业的编译器之类辅助软件。任何一台具有浏览器的个人计算机、手机或平板电脑都能使用。当前个人计算机常用的微软公司的 Windows 操作系统，包括 Windows XP、Windows 7、Windows 8、Windows 10 等；苹果公司的 Mac OS；开源的 Linux 操作系统，包括 ubuntu、Red Hat；等等都能支持。大部分平板电脑和手机也能支持，具有一定的通用性，方便于教学。在教学中可以避免以前教学内容中的图形编程或 VB 编程，学生只学习了编程软件的使用，而忽略了程序设计的算法思想，不能贯彻学习计算思维的重点。

教材采用 Python 作为教学语言之一的原因，并不是笔者认为其更适合于中学教学，而是由于某些高中教材中使用了 Python，为了能有与高中教材进行衔接。Python 语言编写的

程序需要借助于自身特定的软件开发环境（IDE）才能运行。如果使用的操作系统是Windows：当前较稳定的 Windows 版本是 Python 3.8.2 for Windows。请注意 Python 语言当前有 Python 2.x 和 Python 3.x 两种版本，与其他众多的计算机语言不同，高版本 Python 3.x 对以前的版本并不能做到完全兼容。所以一定不要先学习 Python 2.x，再改成 Python 3.x，这样会产生一些问题，不利于学生的学习。Python 3.x 的 IDE 使用并不复杂，教师根据功能菜单应该能完成教材要求的相关操作。

（三）栏目介绍

学习指引：设置情景描述实际问题，尽量贴近学生实际生活，增加趣味性。

知识之窗：相关知识介绍，是完成学习任务必须掌握的内容。

阅读扩展：也是知识介绍，是可选择性的，不学习该部分也能完成本书所有编程任务。

实践活动：设计的课堂活动，用于学生动手操作实践。

思考与探索：在“实践活动”的前提下，引出一些引申的问题，帮助学生理解“实践活动”的内容，或有所提高。

小贴士：一些编程中可能遇到的困难，与“知识之窗”不同的是更偏重于操作中的技巧。

练习提升：即课后作业。

（四）本册书约定

（1）对于 JavaScript 中组合语句标记{、}，{可以放在组合语句的行首，也可以放在上一行的行尾（主要目的是使代码减少一行，便于阅读）。

（2）在流程图中，赋值号用←表示，等号用=表示，不等号用!=表示，分支框是否符合条件标识用 Y、N 表示，逻辑运算用 AND、OR、NOT 表示。

（五）学生学习效果预期

学生接受授课内容进行学习的基本模式是：

【实践活动】→【授课】→【练习提升】

录入程序观察运行结果→分析程序→独立编写程序

（1）基本学习效果：能正确录入程序并运行，得到运行结果。90%+

（2）良好学习效果：在达到基本学习效果的前提下，能阅读程序并通过修改部分语句或参数，灵活使用程序。如得到一个处理一般规模数据的程序，能利用其解决更大规模的数据，达到解决问题的目的。70%+

（3）优秀学习效果：在达到良好学习效果的前提下，能独立编程解决问题。10%

（4）特别优秀（竞赛要求）：在达到优秀学习效果的前提下，能优化程序，提高程序运行效率。1%-

二、评测建议

（一）评测手段

第一章至第三章采用常规类型的评测手段，如：选择、填空、问答等。

第四章至第五章采用常规类型和提交答案类型的评测手段，其中提交答案类型专门为编程题进行评测。重结果不重过程，以解决问题，提交答案为目标。

1. 关于提交答案评测方式的说明

提交答案类型题目的评测是黑箱的，然而以前的黑箱似乎显得相对较狭义。提交答案类型中黑箱的涵义得到了进一步拓展，更突出了对答案结果的唯一追求。过程，或者说是程序，就显得无足轻重了。这也彻底解决了当今程序设计中最难解决的问题，即如何对几近无限种计算机语言编写的程序进行评测。用一句话来概括解决结果提交类问题的原则就是：永远提醒自己，所需要的只是在最短的时间内得到正确结果，而不是过程。

2. 提交答案类型题目样例

有一些加法计算的任务需要完成。需要计算的数据在 10 个文本文件（输入文件）中，需要将这些数据进行加法计算，计算结果放在 10 个文本文件（答案文件）中，作为答案提交。文件名称及格式要求如下。

提供给你的输入文件包括若干行（每个文件的具体行数在表格中说明），每行两个整数，中间用一个空格隔开。

提交的答案文件也包括若干行（必须与对应输入文件一致），每行一个整数，为输入文件对应行中两个整数的和。

10 个输入文件的具体信息如表 1 所示。

表 1　10 个输入文件的具体信息

序号	输入文件名	答案文件名	文件行数	整数范围 (x 表示整数)
1	in1.txt	ans1.txt	1	0<=x<=100
2	in2.txt	ans2.txt	3	0<=x<=100
3	in3.txt	ans3.txt	100	0<=x<=100
4	in4.txt	ans4.txt	200	0<=x<=100
5	in5.txt	ans5.txt	300	0<=x<=100
6	in6.txt	ans6.txt	3000	0<=x<=100
7	in7.txt	ans7.txt	5000	0<=x<=100
8	in8.txt	ans8.txt	10000	0<=x<=100
9	in9.txt	ans9.txt	1	$0<=x<10^{99}$
10	in10.txt	ans10.txt	100	$0<=x<10^{99}$

提示：

可以采用多种方法进行计算，利用 Python 或 JavaScript 编程序完成是一个较好的手段，也可以用其他计算机语言（如 VBScript、C/C++、JAVA 等）编写程序来解决本问题。如果写程序有困难，也可以通过手工计算、计算器计算等方法来解决问题，当然，那样工作量会很大。in9.txt，in10.txt 两个文件中的数据特别大，超出了默认的整数范围，需要想办法解决。

注意：

提交的答案文件必须是纯文本文件（ASCII 码文件），不能是 Word、Excel 等格式的文件。答案文件命名必须按照表格中的要求来命名，答案文件的格式必须符合题目要求。本题评测由机器评分，不符合要求的文件不能得分。

设计思路：

本题考察利用信息技术手段解决数学计算问题的能力。可以利用的手段如下。

（1）手工计算。（计算正确可得 1、2、6 测试点的分数）

（2）利用计算器计算。（计算正确可得 1、2、3 测试点的分数）

（3）利用 excel 通过函数计算。（计算正确可得 1、2、3、4 测试点的分数）

（4）利用循环写程序计算。（计算正确可得 1、2、3、4、5 测试点的分数）

（5）利用高精度算法写程序计算。（满分）

提交答案类型问题的应用不算广泛，但其以新的模式、效率观、时空观，以及因题型而产生的解题技巧、解题策略，和对各种方法（不仅限于编程）的鼓励，对学生更加深入、更加全面的考察而受到青睐，相信能够得到一定的传播并发展，以至成为当程序设计能力考察中的一种重要题型。