第5章 Python编程基础

Python的编程格式丰富多样，本章将系统介绍Python的输入输出、运算符、表达式、控制结构以及函数等内容，主要包括以下任务。

任务一：Python输入与输出应用。

任务二：Python运算符与表达式。

任务三：Python的分支应用。

任务四：Python的循环应用。

任务五：Python函数应用。

本章教学目标

了解Python的基本语法，掌握基本的Python编程，能够独立动手编写程序。

5.1 Python输入与输出应用

本章主要介绍Python的编程。Python的语法结构非常丰富，初学者在学习Python时建议以多学多练为主。本节主要介绍Python的注释和常用的输入、输出函数的使用。

本章以制作一个简单的Python学习系统的完整过程为例，学习Python的编程。首先，简单地分析Python学习系统有哪些功能，为了使后续的代码更简洁，Python学习系统的主要功能仅包括实现播放视频、做练习题、自由练习。随着读者编程能力的提升，Python学习系统可以扩展更多功能。

5.1.1 Python的基本语法

本节主要介绍Python的注释、内置数据类型以及变量的概念和应用。

1. 注释

任何计算机的编程语言都会存在一种非常特殊的文字，计算机不会去识别也不会执行它，但是这段内容可以帮助读者们看懂这个程序是做什么的、怎么实现的，这样的内容为注释。好的注释可以提升代码的可读性。

1）井号注释

井号注释将“#”放在需要注释的文字开头，以便Python解释器识别并跳过该行“#”

后面的文字。

现在读者可以尝试将 4.4.1 节中提到的第一个程序，加入井号注释。“#”表示从“#”开始到这行结束的文字是注释。

例 1，如图 5.1 所示，参见“源代码/ch5/1.hello.py”。

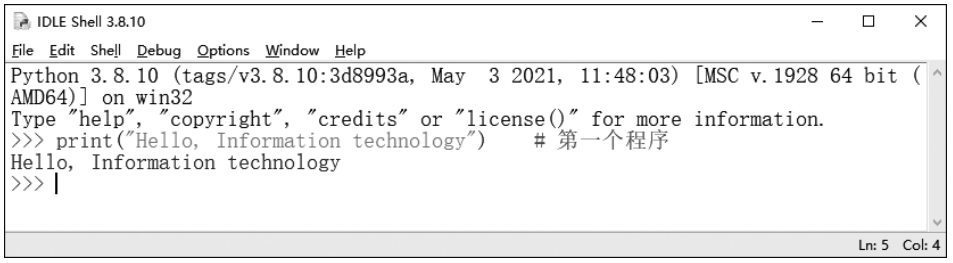

图 5.1 运行语句带注释

2）三引号注释

Python 中还可以使用三单引号表示多行文本的文档注释或表示多行文本字符串充当代码的引用对象。一对三单引号（'''... '''）或者一对三双引号（"""..."""）之间的内容不属于任何语句，也会被解释器认为是注释。

例 2，参见“源代码/ch5/2.comment.py”。

```
1.  '''
2.  This is Multiline String.
3.  It may be programming comments.
4.  '''
5.  print("多行文本注释,可以充当程序代码文档注释")
6.  s ='''
7.  Infomation of Python
8.  Language: Python
9.  characteristic: simple, explicit
10. '''
11. print(s)
```

在这段代码中，程序第 1～4 行是通过三（单）引号括起来的一段文字，这段文字不属于任何语句，因此 Python 的解释器认为它是一段注释。

而程序第 6～10 行里也存在三单引号括起来的一段文字，而这段文字赋值给了变量 s，此时这段文字在 Python 解释器看来，它不是注释，而是一个变量 s 的引用对象。因此，程序第 11 行可以输出这个 s 变量，即输出了这段文字，如图 5.2 所示。

```
========== RESTART: E:\源代码\ch5\2.comment.py ==========
===
多行文本注释，可以充当程序代码文档注释

Infomation of Python
Language: Python
characteristic: simple, explicit
```

图 5.2 三（单）引号注释代码运行结果

由此可以得出，三引号(包括三单引号和三双引号)可以用于程序的注释；它也可以嵌入语句中，充当多行字符串的引用。

2. 内置数据类型

Python的数据类型是由赋值决定的，但是Python为强类型语言，因此需要学习Python的数据类型。下面介绍Python的内置数据类型，Python的内置数据类型通常是基本数据类型。

1) 数字

Python提供整数类型、浮点数类型以及复数类型。

整数类型：包含0和正负整数，可以用十进制、二进制、八进制、十六进制表示。例如5468(十进制)、0x6e(十六进制)。

浮点数类型：浮点数可以用定点小数或科学记数法表示。例如23.5(定点小数)、1.6e8(科学记数法)。

复数类型：Python是支持复数类型的，并可以支持复数计算。Python语言中用j表示$\sqrt{-1}$，例如：6+3j(复数的实数为6，复数的虚数为3j)。

2) 字符串

字符串通常用单引号、双引号或三单(双)引号作为界定符，内容可以是字母、数字或符号等，例如'a'、'hello'、'123'、"aaa@bbb.ccc"。

3) 布尔型

布尔型数据只有两个：True和False，通常是逻辑运算或关系运算的结果，也可以是成员运算或身份运算的结果。例如，3>5的结果为False。

4) 空类型

空类型就是空值：None。这个类型可以自动转换为任何其他类型。

3. Python变量

Python不需要事先声明变量名和类型，只需要通过赋值即可创建各种类型的变量，Python的这种特点直接反映它是一种动态类型语言。

Python的变量是有类型的，而且是强类型，这意味着如果希望将字符串转换为整数，必须通过int()函数进行转换。Python的变量通过赋值语句后面的表达式计算出的值决定变量的类型。一旦类型确定了，那么将不能随意改变。

例3，参见“源代码/ch5/3.variable.py”。

```
1.  >>>t =100              #将100赋值给t
2.  >>>t
3.  100
4.  >>>type(t)             #显示t的类型
5.  <class 'int'>
```

例如将整数100赋值给t，那么，t这个变量的类型就确定为整数类型了。使用内置函数type()可以查看变量t的类型。

Python的变量是存储在内存中的。在内存中，Python的变量是作为一个地址去访问

变量的值，如何显示出变量的地址？可以使用内置函数 id()显示变量的值。

```
>>>id(t)              #t 的内存地址
140715236139888
>>>id(k)              #K 的内存地址
2170771568656
>>>id(t) ==id(k)     #比较 K 和 t 的内存地址是否相同
False
```

请思考，如果要判断 t 和 k 的值是否相等，代码如何写？

5.1.2 print 输出应用

Python 使用内置 print 函数进行输出操作，在 Python2 中 print 不是函数，但是却履行着函数的职责。Python3 彻底将 print 改变成了函数。

要输出一段文字，可以将文字放入 print()的括号内，即文字为函数的参数。

下面实现的代码是在本章提到的 Python 学习系统的菜单的输出。输出菜单让用户可以看到 Python 学习系统有哪些功能。

Python 学习系统需要展示给使用者的菜单提示如下。

```
欢迎使用 Python 学习系统
请选择以下功能数字
1.播放学习视频
2.做练习题
3.自由练习
4.结束
```

代码可以这样完成(用其他的完成方式也是可以的，欢迎读者用各种方式完成这样的输出)。

例 4，参见“源代码/ch5/4.print.py”。

```
welcome ='''
欢迎使用 Python 学习系统
请选择以下功能数字
1.播放学习视频
2.做练习题
3.自由练习
4.结束
'''
print(welcome)          #打印多行字符串
```

5.1.3 input 输入应用

在程序运行时，常常需要用户给程序提供用于计算、判断的参数，这时需要采用 input()函数，让用户通过键盘来输入参数。input()的用法如下。

```
variable =input()
```

或者

```
variable =input('prompt line')
```

input 函数返回键盘输入的字符串；因此，variable 的数据类型是字符串，它不能直接参与数字运算，即使输入的是数字。input()函数可以带参数，也可以不带参数。input()函数的参数会输出到屏幕，提示用户输入合适的内容，因此，这个参数通常是字符串类型。

前面提到了 Python 学习系统的菜单输出界面，已经成功地展示了各项功能的菜单，现在用户需要选择相应的菜单，又应该如何做呢？

使用 input 函数来输入信息，只需要增加以下语句。

```
1. c =input("你的选择是：")                    #提示用户从键盘输入，并返回键盘输入的字符串
2. print("你选择的是菜单第" +c +"项")          #打印用户输入的提示
```

注意，这里虽然输入的变量 c 是数字，但是它是字符串类型，如果希望将它当作数字来处理，还需要强制转换它的类型，可以为 int(c)。

上面的例子存在问题。如果使用者无法理解这段提示，或者故意捣乱，输入的是 5 而非 1～4，这时，程序运行结果错误，如图 5.3 所示。

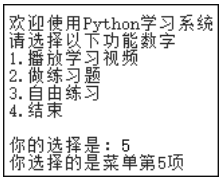

图 5.3　例 4 的运行结果

事实上，Python 学习系统没有菜单第 5 项，这时程序应该提示错误，但是程序还是“傻傻地”正常地运行。这种情况将在 5.3.1 节进一步改进。

例 5，参见“源代码/ch5/5.input.py”。

```
1.  welcome ='''
2.  欢迎使用 Python 学习系统
3.  请选择以下功能数字
4.  1.播放学习视频
5.  2.做练习题
6.  3.自由练习
7.  4.结束
8.  '''
9.  print(welcome)
10.  c =input("你的选择是：")
11.  print("你选择的是菜单第" +c +"项")          #打印用户输入的提示
```

5.2 Python 运算符与表达式

Python 语言提供了丰富而且简单的运算符。最主要、最常见的运算符主要有以下几类。

(1) 算术运算符：主要是对两个对象进行算术运算，其运算逻辑与数学中的计算概念相似。

(2) 赋值运算符：赋值运算符是编程开发中最常用的运算符，即对一个对象进行赋值，将运算符右侧的值赋值给左侧的变量。

(3) 比较运算符：比较运算符主要用于对两个对象进行比较，其对象可以是数值也可以是字符串。

(4) 逻辑运算符：用于逻辑运算的符号，一般用于判断两个对象的与、或、非的结果，返回一个布尔值，其运算原理与数学中的逻辑运算相同。

(5) 位运算符：位运算符对 Python 对象按照二进制方式的每一位的值进行操作，例如，与运算、或运算、移位运算等。

(6) 成员运算与身份运算：成员运算判断两个对象是否存在包含关系；身份运算判断是否引用的是同一对象。

当然 Python 还有其他类型的运算符，在后面会进一步学习。为了让读者对 Python 的运算符有初步的认识，现在通过一个案例展示 Python 运算符的运用。

5.2.1 运算符与表达式案例

Python 表达式是由变量、常量、运算符、函数等组成的。它主要帮助程序做相应数字的计算、逻辑判断、关系成员判断、对象判别等。下面看看 Python 的运算符和表达式到底能做什么？

例如，一元二次方程。是否存在两个不相等的实数根、两个相等的实数根或不存在实根是通过式(5.1)的 Δ 值是否大于 0、等于 0、小于 0 来判别的。

$$\Delta = b^2 + 4ac \tag{5.1}$$

求解这两个根需要使用式(5.2)

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a} \tag{5.2}$$

利用 Python 计算一元二次方程的根，可以这样写 Python 程序。

例 6，参见“源代码/ch5/6.express.py”。

```
1.  import math
2.  a=1                              #赋值参数 a
3.  b=-3                             #赋值参数 b
4.  c=2                              #赋值参数 c
5.  delta =b**2-4*a*c                #计算 delta
```

```
6.  print(delta)                              #打印 delta 的值
7.  x1=(-b+math.sqrt(delta))/(2*a)            #计算 x1
8.  x2=(-b-math.sqrt(delta))/(2*a)            #计算 x2
9.  print(x1,x2)                              #打印 x1,x2 的值
```

此程序因为使用了标准库 math 模块，而 math 模块并不是默认导入的，需要手动导入，因此程序第 1 行就利用 import 语句进行导入。import 语句的语法如下。

```
import  模块名 [as 别名]
```

或

```
from 模块名 import  函数名
```

在例 6 中，直接导入了 math 模块，而没有给 math 模块定义别名。当然可以定义别名，例如，import math as sx。如果这样做了，那么程序第 7～8 行的 math 都要替换为 sx。

在程序中，仅使用了 sqrt()函数，也可以导入从整个模块变为某一个函数，例如 from math import sqrt，这样导入 sqrt()函数后，程序第 7～8 行就要去掉“math.”。

程序第 2～4 行是对一元二次方程的 a、b、c 三个变量进行赋值，a 代表二次项系数，b 代表一次项系数，c 代表常数。

程序第 5 行计算 Δ 的值，这个程序仅显示出了 Δ 的值，并没有做判断。实际上，如果需要判断它是否是实数根，需要如何写这样的表达式呢？只需要将第 6～9 行进行修改，详见 5.3.2～5.3.4 节。

程序第 7～8 行则是用公式计算 x1 和 x2 的值。

在程序最后，记得要让计算机将计算的结果输出，这样程序才算真正地完成了。程序的最后一行输出 x1 和 x2 的值。

如果需要更多的运算符及其功能的详细介绍，可以查看 5.2.2～5.2.5 节。

5.2.2　Python 算术和赋值运算符

Python 吸取了各种语言的优点，运算符非常丰富，甚至还有其他语言没有的运算符。

1. 算术运算符

算术运算符专门用来处理数学的计算问题，当然如果遇到复杂的计算还需要使用 math 模块中的函数，例如 sin、cos 这些函数。

Python 的算术运算符比其他语言丰富，功能强大，如表 5.1 所示。

表 5.1　算术运算符

运算符	描　　述	实　　例
＋	加，两个对象相加	a ＋ b(a＝10，b＝20)输出结果 30，"hello"＋"world"计算结果"hello world"
－	减，前一个对象减后一个对象	a － b(a＝10，b＝20)输出结果－10

续表

运算符	描　　述	实　　例
*	乘，两个对象相乘，或字符串重复多次	a*b(a=10,b=20)输出结果 200，"a" * 3 计算结果"aaa"
/	除	b / a(a=10,b=20)输出结果 2
%	取模，返回除法的余数	b % a(a=10,b=20)输出结果 0
**	幂	a**b(a=10,b=20)输出结果 100000000000000000000
//	整除，返回商的整数部分	9 // 2 计算结果 4，5.0 // 3 计算结果 1.0

上述脚本可以在 IDLE 的交互式界面中运行尝试体验。

例 7-1，（包括 5.2 节所有的 shell 代码），参见“源代码/ch5/7.operation.py”。

```
>>>a=10
>>>b=20
>>>a+b
30
>>>a-b
-10
>>>a*b
200
>>>b/a
2.0
>>>b%a
0
>>>a**b                    #a 的 b 次方
100000000000000000000
>>>9//2                    #9 整除 2 的商
4
>>>5.0//3                  #5.0 整除 2 的商
1.0
>>>"hello"+"world"         #字符串相连
'helloworld'
>>>"a"*3                   #字符串重复
'aaa'
```

2. 赋值运算符

Python 的赋值运算符通常用来对变量赋值，Python 提供了赋值运算符，详细如表 5.2 所示。

表 5.2　赋值运算符

运 算 符	描　述	实　　例
=	赋值	c = a + b
+=	加法赋值	c += a 相当于 c = c + a

续表

运 算 符	描　述	实　　例
－＝	减法赋值	c －＝ b 相当于 c ＝ c － b
＊＝	乘法赋值	c ＊＝ a 相当于 c ＝ c ＊ a
/＝	除法赋值	c /＝ b 相当于 c ＝ c / b
%＝	取模赋值	c %＝ a 相当于 c ＝ c % a
＊＊＝	幂赋值	c ＊＊＝ a 相当于 c ＝ c ＊＊ a
//＝	整除赋值	c //＝ a 相当于 c ＝ c // a

例 7-2，参见“源代码/ch5/7.operation.py”。

```
1.  >>>c =a +b
2.  >>>c
3.  30
4.  >>>c +=a            #将 c+a 的结果赋值给 c
5.  >>>c
6.  40
7.  >>>c -=b            #将 c-b 的结果赋值给 c
8.  >>>c
9.  20
10. >>>c *=a            #将 c*a 的结果赋值给 c
11.  >>>c
12. 200
13. >>>c /=b            #将 c/b 的结果赋值给 c
14. >>>c
15. 10.0
16. >>>c %=a            #将 c%a 的结果赋值给 c
17. >>>c
18. 0
19. >>>c=10
20. >>>c **=a           #将 c**a 的结果赋值给 c
21.  >>>c
22. 10000000000
23. >>>c //=a           #将 c//a 的结果赋值给 c
24. >>>c
25. 1000000000
```

5.2.3　Python 比较运算符

Python 的比较运算符通常是用来比较两个变量或对象(变量的类型可能是数字、字母对象等)的大于、小于、等于等关系的运算符，详细如表 5.3 所示。

表 5.3 比较运算符

运算符	描　　述	实　例
==	判等。这里是双等号区分赋值的等于，其含义也是判断两个对象是否相等	(a == b)
!=	不等，比较两个对象是否不相等	(a != b)
>	大于	(a > b)
<	小于	(a < b)
>=	大于或等于	(a >= b)
<=	小于或等于	(a <= b)

例如：3>5 比较大小，所有人都会知道这是不正确，因此这个表达式返回的值就是 False，构造出一个布尔(Boolean)值，Boolean 值只有 True 和 False。

例 7-3，参见“源代码/ch5/7.operation.py”。

```
1.  a=10
2.  b=20
3.  print(a ==b)          #输出 False
4.  print(a !=b)          #输出 True
5.  print(a >b)           #输出 False
6.  print(a <b)           #输出 True
7.  print(a >=b)          #输出 False
8.  print(a <=b)          #输出 True
```

5.2.4 Python 逻辑运算符

Python 的逻辑运算符，通常用来处理多个 Boolean 型变量或常量之间的运算，逻辑运算符的使用如表 5.4 所示。

表 5.4 逻辑运算符

运算符	描　　述	实　例
and	与。如果 x 为 False，那么返回 False；如果 x 为 True，y 为 False，结果依然 False；如果两个都为 True，则为 True	x and y
or	或。如果 x 为 True，那么返回 True；如果 x 为 False，y 为 True，结果依然 True；如果两个都为 False，则返回 False	x or y
not	非。如果 y 为 False，那么返回 True；如果 y 为 True，那么返回 False	not y

参与 and 运算的两个变量 x 和 y 通常是一个 Boolean 型的变量，而 and 表示两个变量值只有都为 True 时，这个表达式的值才为 True，因此，这里简单概括“一假全假，全真才真”。

而 or 运算的两个变量 x 和 y 只要有一个为 True 即可以返回 True，所以，简单概括“一真全真，全假才假”。