第 3 章 数据的输入与输出

CHAPTER 3

数据的输入与输出是程序与用户交互的主要方式。用户通过输入设备向程序提供数据和信息的行为称为输入；程序将信息或运行结果输出到输出设备的行为称为输出。在 Python 中，输入与输出操作通过一系列内置函数得以实现，这些函数为开发者提供了便捷且灵活的交互方法。input() 函数和 print() 函数是 Python 语言中使用较为广泛的输入 / 输出内置函数。开发者应注意，Python 代码中必须使用英文标点符号，这是因为 Python 的语法规定和大多数编程语言一样，使用英文标点符号。

尽管 Python 默认采用 ASCII 码，目前主流的计算机操作系统多采用汉字国标扩展码（Chinese Character GB Extended Code，GBK）或 UTF-8 编码，这些编码均基于英文字符集设计，不能识别中文标点符号。为降低中文母语者学习编程的语言门槛，促进编程教育普及和本地化应用，开发支持中文的编程工具和资源是有益的。正如我国完全自主研发的天宫空间站在操作面板上全部采用中文一样，未来随着中国人自主设计中文操作系统，编译器和编程语言的发展，中文和中文标点符号在编程中的使用将更加普及。

学习目标

（1）理解程序中数据输入与输出的概念。

（2）熟练掌握 input() 函数和 print() 函数的使用方法。

学习重点

（1）学习并掌握 input() 函数的实际应用。

（2）学习并掌握 print() 函数多种格式的实际应用。

学习难点

print() 函数和 format() 方法均支持多参数输入，参数可涵盖输出内容、分隔符、结束符等，可用于格式化输出，需要熟练掌握 print() 函数和 format() 方法的用法。

3.1 输入函数 input()

视频讲解

开发人员利用 input() 函数可通过键盘获取输入数据，输入的数据类型可以是字符串、整数或浮点数等类型。当程序执行到 input() 函数时将暂停运行，等待用户完成输入操作。用户输入的数据以字符串类型保存在指定的变量中；当输入的数据需要按照数值类型处理时，可以通过 int() 函数将字符串类型转换为整数型，或者通过 float() 函数将

字符串类型转换为浮点型，或者通过 eval() 函数将字符串解析为数学表达式并返回其计算结果。

1. input() 函数的基本格式

基本语法格式如下所示。

```
input([ 提示信息 ])
```

其中，[提示信息] 是可选项，圆括号 () 不能省略，出现的提示信息要使用一对单引号（''），或者一对双引号（" "），或者一对三引号（''' '''）作为定界符，input() 函数的返回值是字符串。示例代码如下所示。

```
>>> x = input('请输入第一个数：')
请输入第一个数：5
>>> y = input('请输入第二个数：')
请输入第二个数：8
>>> x + y              # 使用 "+" 运算符实现两个字符串的连接，得到一个新的字符串
'58'
>>> x = int(input('请输入第一个数：'))
请输入第一个数：5
>>> y = int(input('请输入第二个数：'))
请输入第二个数：8
>>> x + y
13
>>> x = input('请输入一个表达式 :')
请输入一个表达式 :5+8
>>> x
'5+8'
>>> x = eval(input('请输入一个表达式 :'))
请输入一个表达式 :5+8
>>> x
13
```

视频讲解

2. input() 函数的扩展格式

基本语法格式如下所示。

```
input([ 提示信息 ]).split([ 分隔字符 ])
```

其中，split() 的参数如果为空，则默认输入时使用空格作为输入字符串的分隔符，示例代码如下所示。

```
>>> a,b,c = input("请输入 3 个数：").split()      # 使用空格作为输入数据的分隔符
请输入 3 个数：3 4 5
>>> a,b,c = input("请输入 3 个数：").split('-')  # 使用 '-' 作为输入数据的分隔符
请输入 3 个数：1-2-3
>>> a,b,c = input("请输入 3 个数：").split('*')  # 使用 '*' 作为输入数据的分隔符
请输入 3 个数：3*6*9
```

3.2　输出函数 print()

print() 函数在 Python 中用于将数据输出到输出设备（如显示器）中，并能根据提供的参数定制输出格式，下面列举其常见用法。


视频讲解

1. print() 函数的基本格式

print() 函数在输出提示信息时需要使用一对单引号（''），或者一对双引号（" "），或者一对三引号（''' '''）作为定界符，示例代码如下所示。

```
>>> print('伟大祖国')
伟大祖国
>>> print("美好未来")
美好未来
```

使用 print() 函数输出多个字符串时，可以用逗号进行分隔，各字符串将在输出中以空格分隔，示例代码如下所示。

```
>>> print('勤劳', '勇敢', '智慧')
勤劳　勇敢　智慧
```

print() 函数也可以用加号连接字符串，输出时字符串前后紧密相连。示例代码如下所示。

```
>>> print('爱国' + '敬业' + '诚信' + '友善')
爱国敬业诚信友善
```

print() 函数在输出数值类型数据时不加定界符，该函数也可用来进行数值之间的算术运算。示例代码如下所示。

```
>>> print(10)
10
>>> print(20 + 30)
50
```

2. print() 函数的扩充格式

基本语法格式如下所示。

```
print(<多个参数>, sep = ' ', end = '\n')
```

其中，<多个参数>表示允许一次性输出多个数据对象，当输出多个对象时，各对象之间应以逗号作为分隔符。参数 sep 指定多个对象之间的分隔符，其默认值是单个空格。参数 end 用以定义输出序列的结束标记，其默认值是换行符 \n，也可根据需要替换成其他字符。示例代码如下所示。

```
>>> print('未来可期', end = '#')
未来可期#
>>> print('惟愿', '山河锦绣','国泰民安', sep = '@')
```

```
惟愿 @ 山河锦绣 @ 国泰民安
>>> print(' 惟愿 ', ' 和顺致祥 ',' 幸福美满 ', sep = '@',end = '#')
惟愿 @ 和顺致祥 @ 幸福美满 #
```

3. print() 函数的字符串格式化方法

当开发者需要输出非字符串类型的内容时，就需要使用字符串格式化方法。Python 的字符串格式化有多种方法，其中一种方法是在 print() 函数中将 % 作为占位符指定数据或变量在字符串中的插入位置，并结合格式字符定制输出结果。Python 提供了多种格式字符，方便在输出字符串时插入不同类型的变量。常用的格式字符及其说明如表 3-1 所示。


视频讲解

表 3-1　常用的格式字符及其说明

格式字符	说　明
%c	单个字符
%s	字符串
%f 或 %F	浮点数
%e	指数（底数写 e）
%E	指数（底数写 E）
%d 或 %i	十进制整数
%b	二进制整数
%o	八进制整数
%x	十六进制整数
%g	指数（e）或浮点数（根据显示长度）
%G	指数（E）或浮点数（根据显示长度）
%%	字符 %

表 3-1 中的格式字符需要将占位符 % 作为前缀，后面跟上表示数据类型的格式字符。插入对象放在字符串后面紧跟的 %() 内，插入多个对象时需用逗号分隔，插入单个对象时圆括号可以省略。示例代码如下所示。

```
>>> PI = 3.1415926
>>> print("%e %e" % (PI, 1.414))      # 格式化方法输出常量或变量
3.141593e+00 1.414000e+00
>>> print("%f" % PI)                  # 插入单个对象时可以省略圆括号
3.141593
```

下面对常用的字符串格式用法进行举例说明。


视频讲解

1）输出字符串

使用 %s 格式字符来输出字符串，示例代码如下所示。

```
>>> name = " 团结 "
>>> print(" 民族 %s ！ " % name)
民族团结！
```

2）输出整数

使用 %d 格式字符来输出整数，示例代码如下所示。

```
>>> age = 18
>>> print(" 你的年龄是 %d 岁。" % age)
你的年龄是 18 岁。
```

3）输出浮点数

当输出浮点数时，则可以使用 %f 格式字符，示例代码如下所示。

```
>>> PI = 3.1415926
>>> print(" 圆周率的值是 %f " % PI)
圆周率的值是 3.141593
```

4）指定浮点数的输出精度

当需要指定浮点数的输出精度时，可以使用 %.nf 格式字符，其中 n 为要保留的小数位数。也可使用 %.*f 格式字符，此时浮点数的精度将由后续的参数指定，保留精度采用四舍五入的方法。示例代码如下所示。

```
>>> PI = 3.1415926
>>> print(" 圆周率的值是 %.2f " % PI)
圆周率的值是 3.14
>>> print("PI=%.*f" % (3, PI))        # 保留 3 位小数
PI=3.142
```

5）指定最小字符宽度

视频讲解

若需要设定浮点数输出时的总字符宽度，应使用 %m.nf 格式字符，其中 m 为最小字符宽度位数，n 为输出精度。若输出的位数小于 m 设定的最小宽度，系统将在字符串左侧填充空格以补全；若输出位数大于 m 设定的最小宽度，则直接输出实际结果。也可使用 %*.nf 格式字符，此时浮点数的最小字符宽度将由后续的参数指定。需特别指出，在计算字符宽度时，小数点也占一位。示例代码如下所示。

```
>>> PI = 3.141592653
>>> print('%10.3f' % PI)    # 实际输出字符数为 5，最小字符宽度为 10，字符串左侧以
                            # 空格补全
     3.142
>>> print('%2.3f'%PI)       # 实际输出字符数为 5，最小字符宽度为 2，输出实际字符串
     3.142
>>> print("PI=%*.3f"%(10,PI))        # 最小字符宽度为 10
PI=     3.142
```

6）左右对齐

视频讲解

输出浮点数时，默认采用右对齐的方法，%－f 表示左对齐，%＋f 表示在数值前要加上正负号，%0m.nf 表示若位数不够则用 0 填充，示例代码如下所示。

```
>>> PI=3.1415926
>>> print('%-10.3f' %PI)
3.142
>>> PI=3.1415926
>>> print('%+f' % PI)
```

```
+3.141593                # 参数 f 的默认精度为 6 位小数
>>> PI=3.1415926
>>> print('%010.3f'%PI)
000003.142               # 最小字符宽度为 10，前面的 0 表示若位数不够则用 0 进行填充
```

4. print() 函数的转义字符

转义字符以 \ 开始，如 \n 表示换行，\t 表示制表符，\r 表示回车，\f 表示换页等，其他常见的转义字符示例代码如下所示。

```
>>> print('I\'m ok.')
I'm ok.
>>> print('I\'m learning\nPython.')
I'm learning
Python.
>>> print('\\\n\\')
\
\
>>> print('\\\t\\')
\       \
>>> print(r'\\\t\\')     # 允许用 r'' 表示内部的字符串默认不转义
\\\t\\
```


视频讲解

5. print() 函数的 f-string 格式化方法

Python 3.6 及以上版本提供了 f-string 字符串格式化方法，这种新的字符串格式化方法使得字符串的格式化更加直观、简洁和高效。f-string 通过在字符串前加上字母 f 或 F 来标识，允许用户在字符串中嵌入表达式。

f-string 格式化方法的基本格式如下所示。

```
f " < 字符串 > { < 表达式 > } "
```

其中，{ < 表达式 > } 指定表达式在字符串的插入位置；若表达式为算术运算，则将计算结果直接填入到“{}”内。示例代码如下所示。

```
>>> n = "初心"
>>> m = "使命"
>>> print(f " 不忘 {n}，牢记 {m}")
不忘初心，牢记使命
>>> d= {"n":"力量","m":"家园"}
>>> print(f" 磅礴 {d['n']}，建设 {d['m']}")
 磅礴力量，建设家园
>>> PI = 3.1415926
>>> print(f"pi = {PI:.2f}")    # f-string 中可使用格式字符
pi = 3.14
>>> print(f"2pi = {2*PI}")     # f-string 中可嵌入算术运算
2pi = 6.2831852
>>> print(f"2pi = {2*PI:.2f}")
2pi = 6.28
```

6. format() 方法

format() 方法也可实现字符串格式化，该方法通过在字符串中使用花括号（{}）作为占位符来指定参数的插入位置，语法更加灵活。format() 方法可以直接被调用，也可以和 print() 函数结合使用，format() 方法可以将参数按索引编号填充到字符串中，也可以在不输入索引编号的情况下按参数出现的默认顺序来填充，并且同一个参数可以填充多次，这是其他格式不具备的优势。需特别指出，索引编号从 0 开始。示例代码如下所示。

```
>>> print("{} {}".format(" 华夏文明 "," 源远流长 ")) # 不设置指定位置，按默认顺序
' 华夏文明 源远流长 '
>>> print("{0} {1}".format(" 斗转星移 ", " 岁月沧桑 "))        # 设置指定位置
' 斗转星移 岁月沧桑 '
>>> print("{1} {0} {0}".format(" 还看今朝 ", " 盛世中国 ") )  # 设置指定位置
' 盛世中国 还看今朝 还看今朝 '
```

在 format() 方法的 {} 中支持多种格式化选项，通过不同的格式化组合可以精准地控制输出格式，这些格式化选项的基本结构和说明如下。

```
'{key : fill| align| sign| width| precision| type}'.format()
```

（1）key 指定了 format() 方法调用参数的索引编号。需特别指出，索引编号从 0 开始，否则运行时系统会报错。示例代码如下所示。

```
>>> '{0} {1}'.format(' 祖国 ',' 伟大 ')
' 祖国伟大 '
>>> '{1} {0}'.format(' 祖国 ',' 伟大 ')
' 伟大祖国 '
>>> '{1} {2}'.format(' 祖国 ',' 伟大 ')          # 索引编号未从 0 开始，系统报错
IndexError: Replacement index 2 out of range for positional args tuple
```

（2）fill 参数用于指定填充符，被省略时默认值为空格。这个参数在实际应用中较少使用。在某些需要数字格式化输出的场景下，会使用该参数设置逗号来分隔数字。按照国际惯例，每 3 位数字用一个逗号进行分隔。示例代码如下所示。

```
>>> print('{:,}'.format(123456789))
123,456,789
```

（3）align 参数用于格式化文本的对齐方式。align 参数提供 3 种对齐方式：“>”表示右对齐，“<”表示左对齐，“^”表示居中对齐。示例代码如下所示。

```
>>> print('{:*<20}'.format(' 左对齐 '))        # 参数 20 表示字符串输出总宽度
左对齐 **********
>>> print('{:*>20}'.format(' 右对齐 '))
********** 右对齐
>>> print('{:*^20}'.format(' 居中 '))
********* 居中 *********
```

（4）sign 参数用于指定是否保留正负号，仅对 format() 中的数值起作用。“+”表示保留正号；“-”表示仅保留负号；空格表示正数留空，负数保留负号。示例代码如下所示。

```
print('{0:+} {0:-} {0:}'.format(123))
print('{0:+} {0:-} {0:}'.format(-123))
```

运行结果如下所示。

```
+123 123 123
-123 -123 -123
```

（5）width 参数用于控制输出的长度。当设定的 width 参数小于 format 中调用值的宽度时，width 参数不生效；反之，会用空格（默认）或指定字符进行填补。例如，如果想要用 0 进行填补，需要在 width 前面添加 0。示例代码如下所示。

```
print('{0:2},{0:5},{0:05}'.format(123))
print('{0:<05},{0:>05},{0:^05}'.format(123))
print('{0:^05},{1:^05}'.format(123,1234))
```

运行结果如下所示。

```
123,  123,00123
12300,00123,01230
01230,12340
```

（6）precision 参数用数字“f”表示数据的精确度，即小数点后的保留位数；如果不加“f”，则表示保留有效数字位数。示例代码如下所示。

```
print('{0:.2f},{0:.7f},{0:.2} '.format(123.456789))
```

运行结果如下所示。

```
123.46,123.4567890,1.2e+02
```

（7）type 参数用于指定数据的格式类型。type 参数支持多种数据类型，常用的参数类型及说明如表 3-2 所示。

表 3-2　常用的参数类型及说明

参数类型	说　明
b	以二进制格式输出
c	将整数转换成对应的 Unicode 字符
d	以十进制输出（默认选项）
o	以八进制输出
x 或 X	以十六进制小写或大写输出
e 或 E	指数标记；使用科学记数法输出，用 e 或 E 来表示指数部分，默认精度为 6
f 或 F	以定点形式输出数值，默认精度为 6
g	对于给定的精度 p≥1，取数值的 p 位有效数字，并以定点或科学记数法输出（默认选项）
G	与 g 相同，当数值过大时使用 E 来表示指数部分
n	与 g 相同，但使用当前环境的分隔符来分隔每 3 位数字
#	以二进制、八进制、十六进制等输出时，加上对应的前导符号
%	百分比标记；使用百分比的形式输出数值，同时设定 f 标记

各参数的使用示例代码如下所示。

```
print("{0:b},{0:c},{0:d},{0:o},{0:x},{0:X}".format(65))
print("{0:b},{0:c},{0:#d},{0:#o},{0:#x},{0:#X},{0:.2%}".format(65))
```

运行结果如下所示。

```
1000001,A,65,101,41,41
1000001,A,65,0o101,0x41,0X41,6500.00%
```

【例 3.1】编写程序，模拟简单的人机对话。

示例代码如下所示。

```
xm=input("爱华：我是机器人爱华，请问您的姓名是？  ")
print("爱华："+xm+"您好！") #等价输出方法 print("爱华：{}您好！".format(xm))
wt=input("爱华：您想问什么？  ")
print("爱华：社会主义核心价值观")
print("      富强、民主、文明、和谐")
print("      自由、平等、公正、法治")
print("      爱国、敬业、诚信、友善")
```

运行结果如下所示。

```
爱华：我是机器人爱华，请问您的姓名是？  小敏
爱华：小敏您好！
爱华：您想问什么？  社会主义核心价值观
爱华：社会主义核心价值观
      富强、民主、文明、和谐
      自由、平等、公正、法治
      爱国、敬业、诚信、友善
```

【例 3.2】编写程序，从键盘输入一个小写字母，输出对应的大写字母。

视频讲解

示例代码如下所示。

```
a=input("请输入一个小写字母：")
print("%c 的大写字母是 "%a, a.upper())
# 内置的 upper() 函数可将字符串中的所有字母转换为大写字母
```

运行结果如下所示。

```
请输入一个小写字母：g
g 的大写字母是 G
```

【例 3.3】编写程序，从键盘输入半径，输出圆的面积并保留两位小数。

示例代码如下所示。

```
r=eval(input("请输入一个圆的半径："))
print("圆的面积是 %.2f"%(3.14*r*r))
```

运行结果如下所示。

```
请输入一个圆的半径：6.5
圆的面积是 132.66
```

【例 3.4】编写程序，模拟一个简单的计算器小程序，计算并输出任意两个数的和、差、积、商。

示例代码如下所示。

```
print("欢迎使用北斗计算器！")
x=eval(input("请输入第一个数："))
y=eval(input("请输入第二个数："))
print("%.2f+%.2f=%.2f"%(x,y,x+y))
print("%.2f-%.2f=%.2f"%(x,y,x-y))
print("%.2f*%.2f=%.2f"%(x,y,x*y))
print("%.2f/%.2f=%.2f"%(x,y,x/y))
```

运行结果如下所示。

```
欢迎使用北斗计算器！
请输入第一个数：25.6
请输入第二个数：39
25.60+39.00=64.60
25.60-39.00=-13.40
25.60*39.00=998.40
25.60/39.00=0.66
```

视频讲解

【例 3.5】编写程序，从键盘输入任意一个三位数，将它逆序输出，如输入 123 后输出 321。

示例代码如下所示。

```
x=int(input("请输入一个三位数："))
a=x//100                          # 三位数的百位数
b=(x//10)%10                      # 三位数的十位数
c=x%10                            # 三位数的个位数
y=c*100+b*10+a                    # 计算三位数的逆序
print("%d 的逆序是 %d"%(x,y))
```

运行结果如下所示。

```
请输入一个三位数：123
123 的逆序是 321
```

小结

本章详细阐述了数据的输入与输出函数，不同数据类型对输入数据和输出结果的具体影响，并对 input() 函数和 print() 函数的各种使用方法进行了深入讲解。

【思政元素融入】

本章强调了理解输入信息的重要性，以及清晰表达输出内容的必要性，这一过程不

仅凸显了有效交互在信息处理中的核心地位，也有助于锻炼学生在实际生活中的沟通与表达能力。Python 中不同数据类型对输入和输出的影响，体现了对数据类型多样性的尊重。print( ) 函数强大的功能和丰富的字符串格式化方法，强调了注重细节和灵活应变在编程工作中的决定性作用。这些内容的学习对于培养严谨的工作作风和全面的专业素养具有积极作用。

习题

一、选择题

1. 执行语句 a＝input("")，当用户输入 2＋3 时，变量 a 的值是 _____。

A. 2　　B. 3　　C. 5　　D. '2+3'

2. 执行语句 a＝input("").split(", ")，当用户想输入 a1, b2, c3 时，以下正确的操作是 _____。

A. a1 b2 c3　　B. a1*b2*c3*　　C. a1, b2, c3　　D. 以上都可以

3. 语句 print("%e＝%f"%(674.5, 674.5)) 的运行结果是 _____。

A. 674.5＝6.745e＋02　　B. 674.500000＝6.745000e＋02

C. 6.745000e＋02＝674.500000　　D. 674.5＝6.745000E＋02

4. 语句 print("%08d%＋3.2f"%(1234, 1234)) 的运行结果是 _____。

A. 00001234＋1234.00　　B. 00001234＋123.40

C. 1234.00＋00001234　　D. 1234.00＋0001234

5. 语句 str1＝"good good study", print("%3.4s%5.4s"%(str1, str1)) 的运行结果是 _____。

A. goo goo　　B. goo good　　C. good goo　　D. good good

6. 语句 print("{2}{1}、{0}、{3}".format(' 张三 ', 89, ' 李四 ', 98)) 的运行结果是 _____。

A. 李四 98、89、98　　B. 张三 98、89、98

C. 张三 89、李四、98　　D. 李四 89、张三、98

7. 语句 print("{}＝0x{:04o}＝0o{:04x}".format(15, 15, 15)) 的运行结果是 _____。

A. 15＝0o0017＝0x000f　　B. 15＝0x0017＝0o000f

C. 15＝0o17＝0xf　　D. 15＝0o15＝0x15

8. 语句 print("{:&>6d}".format(89)) 的运行结果是 _____。

A. 89&&&&　　B. &&&&89　　C. >>>>89　　D. 89 >>>>

9. 语句 print("{:0^6d}".format(89)) 的运行结果是 _____。

A. 008900　　B. 800009　　C. 000089　　D. 890000

10. 语句 print("{:<6}，{:,d} 元 ".format(" 笔记本 ", 89)) 的运行结果是 _____。

A. &&& 笔记本，89 元　　B. 笔记本 &&&，89 元

C. 笔记本，89 元　　D. 笔记本，89 元

二、填空题

1. 语句 print(1, 2, 3, sep＝"*") 的运行结果是 _____。

2. 语句 print(4, 5, 6, sep＝"/", end＝"＋") 的运行结果是 _____。

3. 语句 print("%f＋%f＝"%(8, 9)) 的运行结果是 _____。

4. 语句 print("{}, {}".format(" 年龄 ", 18)) 的运行结果是 _____。
5. 语句 print("{0}, {1}".format(" 年龄 ", 18)) 的运行结果是 _____。
6. 语句 print("{1}, {0}, {1}".format(" 年龄 ", 18)) 的运行结果是 _____。
7. 语句 print("{:X}".format(16)) 的运行结果是 _____。
8. 语句 print("{:*^10}".format(' 中华民族 ')) 的运行结果是 _____。
9. 语句 print("{:-^20}".format('123456')) 的运行结果是 _____。
10. 语句 print("{:<12.5}".format("python!")) 的运行结果是 _____。

三、编程题

1. 程序运行时输入“小华”后，请写出下列语句的运行结果。

```
inta=input（" 请输入：")
stra=" 我爱祖国 "
print(inta,stra)
lista=[1,2," 团结 ",3,4]
print(lista)
print(" 少年强 "," 则国强 ",sep="")
print(" 少年强 "," 则国强 ",sep=",")
print(" 少年强 "," 则国强 ",sep="_-_")
print(" 少年富 ",end="")
print(" 则国富 ")
```

2. 从键盘输入一个大写字母，输出它的小写字母。
3. 从键盘输入球体的半径，输出球体的表面积和体积。
4. 从键盘输入 3 个字符串，输出它们并以 * 来分隔字符串。
5. 从键盘输入 3 个整数，输出它们的和及平均值。平均值要求保留两位小数。
6. 从键盘输入一个十进制整数，输出它对应的八进制数。