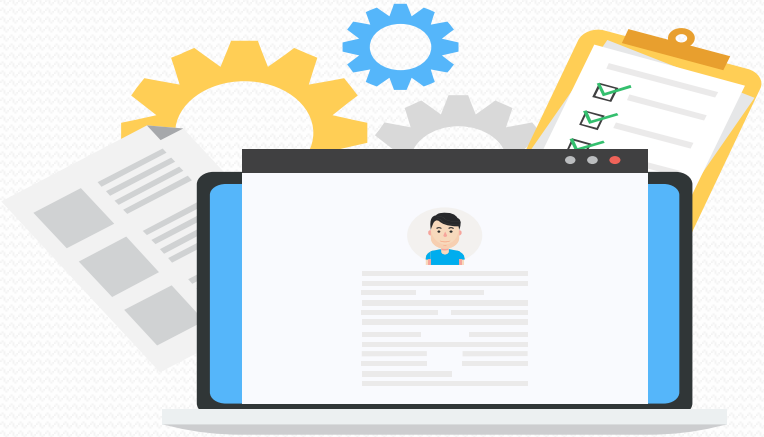
第二章 Python编程基础

Contents



01 Python语言输入与输出

任意计算机程序都应有零个或多个输入，且至少应有一个输出。一般地，计算机程序按照指令位置的先后顺序执行；当遇到分支指令和循环指令时，程序将有条件的跳转到相应指令处执行。本章将介绍按顺序执行的简单Python语言程序，并借助于这类程序讨论Python语言的数据表示方法和基本运算符，然后详细讨论最常用的数据结构——列表和字符串，下面从介绍Python语言程序的输入和输出函数开始。

2.1 Python语言输入与输出

在控制台应用程序工作模式下，Python语言的输入函数为input，输出函数为print。

- 函数input的用法：

  s = input() 或 s=input('提示信息')

- 函数print的用法：

  print()

  print(待输出的一个或多个对象)

  print(待输出的一个或多个对象，sep=' '，end='\n', file=sys.stdout)

2.1 Python语言输入与输出

Python语言输入函数input和输出函数print的用法实例：

```
1    name = input('Please input your name:')
2    print(type(name))
3    print(isinstance(name, str))
4    print(name)
```

第1行调用Python内置函数input将键盘输入的字符串给name，input函数具有提示信息“Please input your name:”。输入了“Yong Zhang”，如果直接回车，将输入空字符。

2.1 Python语言输入与输出

第2行调用Python内置函数type得到name的类型，然后，调用print函数输出这个类型。

第3行print(isinstance(name, str))调用内置函数isinstance判断name是否为str字符串类型，如果是，则返回真；否则返回假。

第4行“print(name)”调用print函数输出name。输入了“Yong Zhang”，这里将输出“Yong Zhang”，可见输入以回车作为结束符。

2.1 Python语言输入与输出

```
6     age = input('Please input your age:')
7     n1 = int(age)
8     n2 = int(age, 10)
9     print(type(n2))
10    s1 = str(n2)
11    s2 = hex(n2)
12    print('My age:' + str(n1), n2, s1, s2, sep=', ', end='.\n')
```

2.1 Python语言输入与输出

第6行调用input函数将键盘输入的字符串赋给age，input函数的参数“Please input your age:”为提示信息。

第7行调用内置函数int将字符串age转化为整数，赋给n1。

第8行展示了完整的int函数的调用形式，这里将以十进制数形式表示的字符串转化为整数，赋给n2。这里的参数10表示age为十进制数组成的字符串。

2.1 Python语言输入与输出

第9行输入n2的类型，将得到“<class 'int'>”，表示n2为整型int，整型int在Python语言中也是一个类。

第10行调用内置函数str将n2转化为字符串，赋给s1。

第11行调用内置函数hex将n2转化为十六进制数，赋给s2。

第12行调用print函数输出结果“My age:28, 28, 28, 0x1c.”。“"My age:' + str(n1)”中的“+”号表示连接两个字符串；print函数可以输出字符串和整数等各种类型的数据；“sep=', '”表示输出的两个对象间用逗号分开；“end='.\n'”表示输出最后一个对象后，将输出一个回车换行。

2.1 Python语言输入与输出

```
14    fd = open('zy0201.txt', mode='w')
15    print(n1, n2, s1, s2, sep=', ', end='.\n', file=fd)
16    fd.close()
```

2.1 Python语言输入与输出

第14行在当前工程所在目录下创建一个文件zy0201.txt，“mode='w'”表示打开方式为“写入”类型，打开的文件对象设为fd。将文件对象fd不再使用时，应执行第16行“fd.close()”关闭它。

第15行使用关键字参数“file=fd”将文件对象作为输出设备，即向文件对象fd中写入信息“28, 28, 28, 0x1c.”。

2.1 Python语言输入与输出

Python语言程序设计的一些规则如下：

（1）Python语言中没有“变量”这种概念，无法像其C语言那样定义一个变量再给变量赋值。因此，也没有变量类型的说法。需要为一些量指定名称时，直接将那些量赋给一些合法的标识符就行，这些标识符的用法类似于“变量”的用法，但含义上有本质的区别，Python中这些标识符并不指代具体的“变量”内存空间。

2.1 Python语言输入与输出

（2）Python中所有数据类型均为类，因此，所有的数据均为对象。所以，Python是完整意义上的面向对象语言。

（3）虽然Python语言中没有变量和变量类型的说法，但是，Python语言具有明确的数据类型。

（4）Python语言使用缩进表示语句间的关系。

2.1 Python语言输入与输出

其他一些规则如下：

（1）注释使用“#”号。任一行中，“#”及其后续的内容均为注释，不会执行。选中多行语句时，按下快捷键“Ctrl + /”可以将选中的语句注释掉。

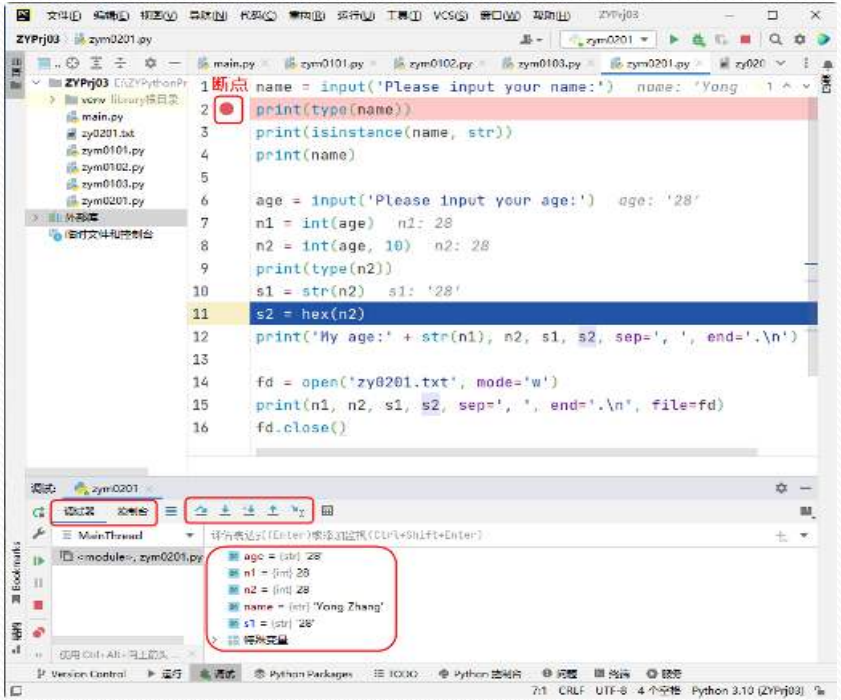
（2）Python语言具有优秀的内存清理机制，程序中不再使用的内存空间，将被自动释放掉。这是所有计算机语言都极力追求的内存管理机制。

2.1 Python语言输入与输出

（3）Python中文件称为模块，比模块更大的概念称为“包”，可以理解为包含了多个文件的目录；比模块更小的概念称为类和函数，类和函数是文件中定义的类型。借助于import可以在当前模块中包含其他的模块，从而实现代码的复用。Python语言的内置模块无需import，可以直接使用这些内置模块及其中的函数。Python语言是实现团队协作编程的最佳选择。

2.1 Python语言输入与输出

程的方式因人而异，但是当程序稍大后，往往需要借助于“调试”功能发现程序的错误。PyCharm提供了优秀的调试功能。图中，单击程序行号所在处，可以为该行语句设置断点。其下方窗口中具有“调试器”和“控制台”选项。



调试工作窗口

2.1 Python语言输入与输出

```
1    import re
2
3    if __name__=='__main__':
4        name = input('Please input your name:')
5        print(type(name))
6        print(isinstance(name, str))
7        print(name)
```

2.1 Python语言输入与输出

第1行使用import装载了re模块，re模块为正则表达式模块，正则表达式是指与字符串匹配相关的规则表达式。

第3行“if __name__=='__main__':”使得第4~20行均为本模块内可执行的语句，当本模块被其他模块调用时，这些语句不被执行。

2.1 Python语言输入与输出

```
9       age = input('Please input your age:')
10      if re.match('^[1-9]\d*$', age) != None:
11          n1 = int(age)
12          n2 = int(age, 10)
13          print(type(n2))
14          s1 = str(n2)
15          s2 = hex(n2)
16          print('My age:' + str(n1), n2, s1, s2, sep=', ', end='.\n')
```

2.1 Python语言输入与输出

第9行“age = input('Please input your age:')”将键盘输入的字符串赋给age。

第10行语句使用了正则表达式判断age字符串中是否只含有数字，且不以数字0开头。这里“re.match”表示调用模块re的match函数，如果它的第一个参数（正则表达式）和第二个参数（字符串）匹配成功，则返回匹配的字符串；否则，返回None。

如果第10行的if语句条件为真，即字符串age中仅包含有效的数字字符，则执行第11~20行。

2.1 Python语言输入与输出

前面的正则表达式“^[1-9]\d*$”表示从字符串的头部（用^表示）开始匹配，一直匹配到字符串的尾部（用$表示），“[1-9]”表示匹配字符串中的一个数字1至9的字符，“\d”可以匹配数字0至9的字符，“*”表示重复匹配0次或多次，“\d*”表示匹配数字0~9的字符0次或多次。因此正则表达式“^[1-9]\d*$”表示匹配的字符串应为“以数字1至9的字符开头，后续字符只能为0至9的数字字符”。

2.1 Python语言输入与输出

```
18          fd = open('zy0201.txt', mode='w')
19          print(n1, n2, s1, s2, sep=', ', end='.\n', file=fd)
20          fd.close()
```

02 Python基本数据类型

2.2 Python基本数据类型

Python语言的基本数据类型包括整数、浮点数和复数等数值类型、布尔类型、字符串类型、字节串类型和空类型等。字符串类型将在第2.5节介绍，空类型只有一个值None，字节串类型为形如“b'Hello'”（Hello可替换为任意ASCII字符或任意8比特表示的字符（扩展ASCII集中的字符））的字符串，布尔类型只有Ture和False两个值。这里重点介绍数值类型的数据类型。

2.2 Python基本数据类型

- Python语言中，数值类型的规则如下：

（1）整数是指十进制整数，为不带小数点的数值，具有无限精度（严格上讲，整数的大小受计算机内存的大小限制）。而二进制、八进制和十六制制的整数通常以字符串的形式表示。例如，3、5、100、-8等均为整数，将这些数作为type函数的参数，将返回“<class ‘int’>”。

2.2 Python基本数据类型

（2）浮点数占8个字节，由于Python语言是用C语言实现的，Python中的浮点数就是C语言中的double类型，存储格式为IEEE-754标准。浮点数是指带有小数点的数，或者是带有指数部分的数，例如，3.、.12、5.3、4e1、2e0等都是浮点数，将这些数作为type函数的参数将返回“<class 'float'>”。

2.2 Python基本数据类型

（3）复数的实部和虚部一定都是浮点数，即使向实部或虚部赋了整数，也自动转化为浮点数。例如，a=complex(3,5)得到复数a，其实部为a.real，虚部分为a.imag，此时，type(a)将返回“<class 'complex'>”，而type(a.real)和type(a.imag)都将返回“<class 'float'>”。

2.2 Python基本数据类型

- Python语言中，复数的常规运算：

（1）求模运算。借助于内置函数abs可以计算一个复数的模，也可以计算一个整数或浮点数的绝对值。abs函数作用于复数和浮点数时返回浮点数，而作用于整数时，返回整数。

（2）求辐角。需要装载cmath包（import cmath），然后，执行“cmath.phase(a)”返回复数a的辐角，当a为正数时，返回0.0；当a为“1j”。

03 Python语言程序结构

2.3 Python运算符

Python语言中，数据的基本处理借助于运算符实现。引用早期的汇编语言的说法，运算符称为操作符，数据称为操作数。严格意义上，Python语言只有单目运算符和双目运算符。运算符具有优先级和结合性（指运算顺序）等属性，在一个（表达式）语句中，先计算优先级高的运算符（及其直接相关的操作数）；再计算优先级较低的运算符；同级别优先级的运算符，按约定的结合性（运算顺序）进行运算，一般为自左向右运算（赋值运算符从右向左）。

2.3.1 算术运算符

- Python语言的算术运算符：

序号	运算符	含义	用法举例
1	+	加法	5+3 得到 8
2	-	减法	5-3 得到 2
3	*	乘法	3*4 得到 12
4	/	除法	5/2 得到 2.5
5	**	乘方	4**2 得到 16
6	%	求余（不适用复数）	7%3 得到 1（可用于浮点数）
7	//	整除（向下取整，不适用复数）	6.5/3.1=2.0

2.3.1 算术运算符

序号	运算符	含义	用法举例
8	+（正）	+x，表示正数	+5 得到 5，+(-5)得到-5
9	-（负）	-x，表示负数	-5 得到-5，-(-5)得到 5
10	abs(x)	求 x 的绝对值（或复数的模）	abs(3+4j)得到 5.0
11	pow(x,y)	求 x^y，即 x**y	pow(4,2)得到 16（整型） pow(4.,2)得到 16.0（浮点型）
12	complex(x,y)	x+yj	complex(3,5)得到(3+5j)
13	c.conjugate()	求复数 c 的共轭复数	3+5j.conjugate()得到(3-5j)
14	int(str)	将 str 转化为整数	int('3')得到 3 int(5.2)得到 5
15	float(str)	将 str 转化为浮点数	float('4.3')得到 4.3 float(5)得到 5.0

2.3.2 位运算符

序号	运算符	含义	用法举例
1	&	按位与	0b110101 & 0b011101 得到 21
2	|	按位或	0b0110101 | 0b011101 得到 61
3	~	按位取反	~0b110101 得到-54
4	^	按位异或	0b110101 ^ 0b011101 得到 40
5	<<	按位左移	0b110101 <<2 得到 212
6	>>	按位右移	0b110101 >>2 得到 13

2.3.3 关系运算符

序号	运算符	含义	用法举例
1	==	等于	3==5得到False
2	!=	不等于	3!=5得到True
3	>	大于	5>3得到True
4	>=	大于等于	5>=3得到True
5	<	小于	5<3得到False
6	<=	小于等于	5<=3得到False

2.3.3 关系运算符

在Python语言中，关系运算符可以直接组合在一起使用，例如，3<=5>4在Python语言中返回True；5>3<8>4将返回真；10>7>5>3也将返回True。尽量避免这种组合使用。关系运算符连接成的表达式称为关系表达式，关系表达式的结果为逻辑值True或False。

2.3.4 关系运算符

- Python语言中逻辑运算符：

序号	运算符	含义	用法举例
1	and	x and y	True and False 得到 False
2	or	x or y	True or False 得到 True
3	not	not x	not True 得到 False

在Python语言中，逻辑运算符可以对整数和浮点数进行操作，并且，将0视为假，非0视为真。