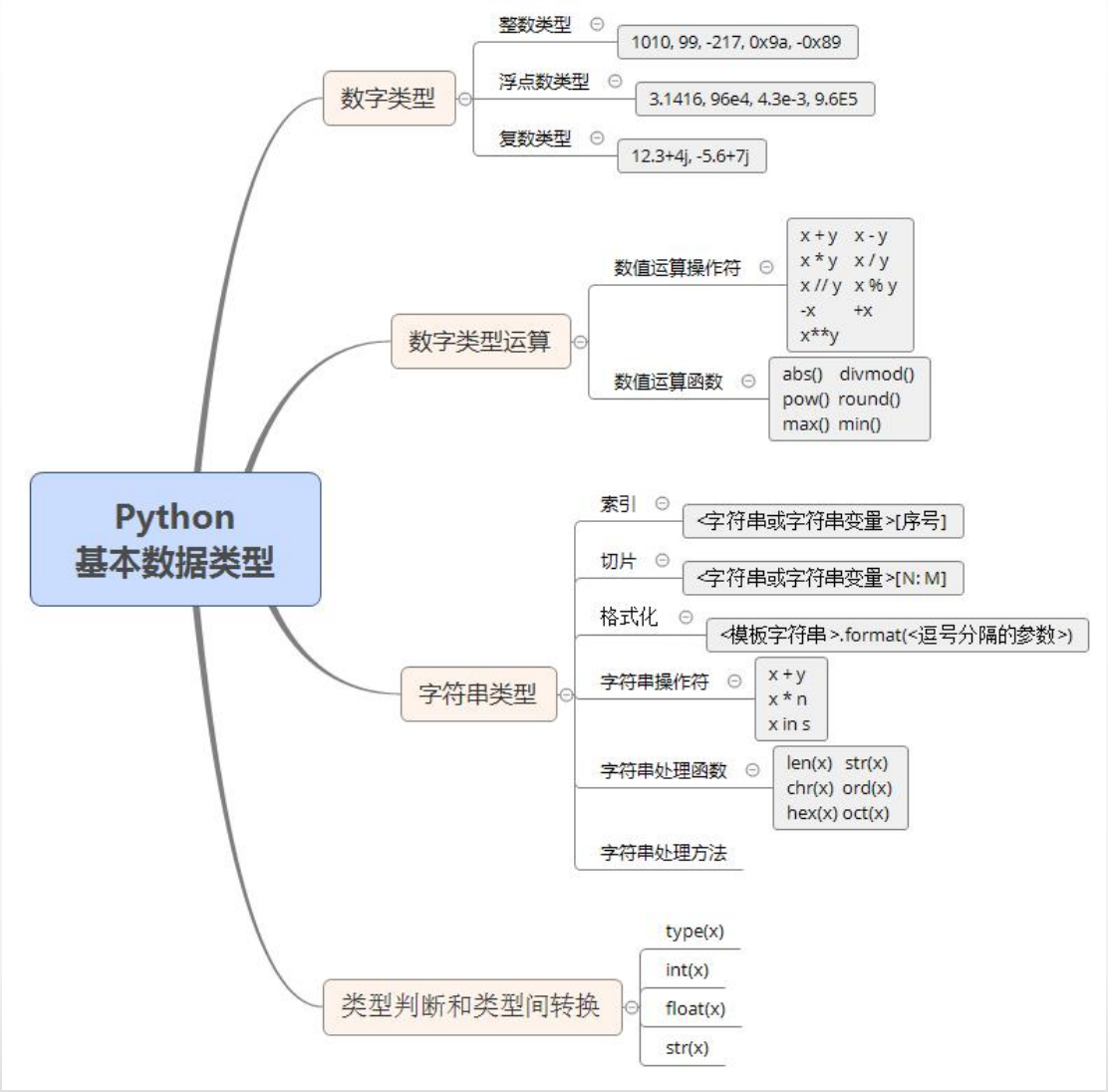
全国计算机等级考试二级教程
Python语言程序设计
(2018年版)

【第3章】
基本数据类型

考纲考点

- 数字类型：整数类型、浮点数类型和复数类型
- 数字类型的运算：数值运算操作符、数值运算函数
- 字符串类型及格式化：索引、切片、基本的format()格式化方法
- 字符串类型的操作：字符串操作符、处理函数和处理方法
- 类型判断和类型间转换

知识导图


Python
基本数据类型
数字类型
整数类型
1010, 99, -217, 0x9a, -0x89
浮点数类型
3.1416, 96e4, 4.3e-3, 9.6E5
复数类型
12.3+4j, -5.6+7j
数字类型运算
数值运算操作符
x + y x - y
x * y x / y
x // y x % y
-x +x
x**y
数值运算函数
abs() divmod()
pow() round()
max() min()
字符串类型
索引
<字符串或字符串变量>[序号]
切片
<字符串或字符串变量>[N: M]
格式化
<模板字符串>.format(<逗号分隔的参数>)
字符串操作符
x + y
x * n
x in s
字符串处理函数
len(x) str(x)
chr(x) ord(x)
hex(x) oct(x)
字符串处理方法
类型判断和类型间转换
type(x)
int(x)
float(x)
str(x)

数字类型

数字类型

- Python语言提供3种数字类型：**整数类型、浮点数类型和复数类型**，分别对应数学中的整数、实数和复数。
- 1010是一个整数类型，10.10是一个浮点数类型，10 + 10j是一个复数类型。

整数类型

- 与数学中的整数概念一致，没有取值范围限制
- 整数类型有4种进制表示：十进制、二进制、八进制和十六进制。默认情况，整数采用十进制，其他进制需要增加引导符号

整数类型

- 示例
  - 1010, -1010, 0b1010, 0o1010, 0x1010

进制种类	引导符号	描述
十进制	无	默认情况，例：1010,-1010
二进制	0b 或 0B	由字符0和1组成，例：0b1010，0B1010
八进制	0o 或 0O	由字符0到7组成，例：0o1010，0O1010
十六进制	0x 或 0X	由字符0到9、a到f或A到F组成，例：0x1010

- 不同进制的整数之间可以直接运算

浮点数类型

- 带有小数点及小数的数字
- Python语言中的浮点数类型必须带有小数部分，小数部分可以是0。例如：1010是整数，1010.0是浮点数。

浮点数类型

- 浮点数有2种表示方法：十进制形式的**一般表示**和**科学计数法**表示。除十进制外，浮点数没有其他进制表示形式。下面是浮点数类型的例子：
- 1010.0, -1010., 1.01e3, -1.01E-3

浮点数类型

- 科学计数法使用字母e或者E作为幂的符号，以10为基数，含义如下：

$$<a>e<b> = a*10^{b}$$

- 上例中，1.01e3值为1010.0；-1.01E-3值为0.00101。

浮点数类型

- Python浮点数类型的数值范围和小数精度受不同计算机系统的限制。除高精度科学计算外的绝大部分运算来说，浮点数类型的数值范围和小数精度足够“可靠”。

```
>>>1234567890.987654321 * 1234567890.987654321
1.524157877457704e+18
>>>9876543210.123456789 / 1234567890.987654321
7.000000066600002
```

复数类型

- 复数类型表示数学中的复数。复数有一个基本单位元素j，叫作“虚数单位”。含有虚数单位的数被称为复数。例如：
- 11.3+4j　　-5.6+7j　　1.23e-4+5.67e+89j

复数类型

- Python语言中，复数可以看作是二元有序实数对（a, b），表示为：a + bj，其中，a是实数部分，简称实部，b是虚数部分，简称虚部。**虚数部分通过后缀“J”或者“j”来表示**。需要注意，当b为1时，1不能省略，即 1j表示复数，而j则表示Python程序中的一个变量。

复数类型

- 复数类型中实部和虚部都是浮点类型，对于复数z，可以用z.real和z.imag分别获得它的实数部分和虚数部分

```
>>>(1.23e4+5.67e4j).real
12300.0
>>>(1.23e4+5.67e4j).imag
56700.0
>>>1.23e4+5.67e4j.imag    # 先获得5.67e4j的虚部，再与1.23e4进行求和计算
69000.0
```

数字类型的运算

数值运算操作符

- Python提供了9个基本的数值运算操作符

操作符	描述
x + y	x与y之和
x - y	x与y之差
x * y	x与y之积
x / y	x与y之商
x // y	x与y之整数商，即：不大于x与y之商的最大整数
x % y	x与y之商的余数，也称为模运算
-x	x的负值，即：x*(-1)
+x	x本身
x**y	x的y次幂，即：x^y

数值运算操作符

- 上标所有二元运算操作符（+、-、*、/、//、%、**）都可以与等号（=）相连，形成增强赋值操作符（+=、-=、*=、/=、//=、%=、**=）。用op表示这些二元运算操作符，增强赋值操作符的用法如下：

x op= y 等价于 x = x op y

```
>>>x = 99
>>>x **=3   # 与x = x**3等价
>>>print(x)
970299
```

数值运算操作符

数值运算可能改变结果的数据类型，类型的改变与运算符有关，有如下基本规则：

- 整数和浮点数混合运算，输出结果是浮点数；
- 整数之间运算，产生结果类型与操作符相关，/运算的结果是浮点数；
- 整数或浮点数与复数运算，输出结果是复数。

数值运算操作符

```
>>>1010/10          # /运算的结果是浮点数
101.0
>>>1010.0//3       # 浮点数与整数运算，产生结果是浮点数
336.0
>>>1010.0 % 3      # 浮点数与整数运算，产生结果是浮点数
2.0
>>>10 - 1 + 1j     # 等价于 10 - (1 + 1j)
(9+2j)
```

数值运算函数

- Python解释器提供了一些内置函数，在这些内置函数之中，有6个函数与数值运算相关

函数	描述
abs(x)	x的绝对值
divmod(x, y)	(x//y, x%y)，输出为二元组形式（也称为元组类型）
pow(x, y[, z])	(x**y)%z，[..]表示该参数可以省略，即：pow(x,y)，它与x**y相同
round(x[, ndigits])	对x四舍五入，保留ndigits位小数。round(x)返回四舍五入的整数值
max(x_1, x_2, …, x_n)	x_1, x_2, …, x_n的最大值，n没有限定
min(x_1, x_2, …, x_n)	x_1, x_2, …, x_n的最小值，n没有限定

字符串类型及格式化

字符串类型

- 字符串是字符的序列表示，根据字符串的内容多少分为单行字符串和多行字符串。
- 单行字符串可以由一对单引号（'）或双引号（"）作为边界来表示，单引号和双引号作用相同。
- 多行字符串可以由一对三单引号（'''）或三双引号（"""）作为边界来表示，两者作用相同。

字符串类型

```
>>>print('这是"单行字符串"')
这是"单行字符串"
>>>print("这是'单行字符串'")
这是'单行字符串'
>>>print("""这是'多行字符串'的第一行
这是'多行字符串'的第二行
""")
这是'多行字符串'的第一行
这是'多行字符串'的第二行
>>>print('''这是"多行字符串"的第一行
这是"多行字符串"的第二行
''')
这是"多行字符串"的第一行
这是"多行字符串"的第二行
```

字符串类型

- Python语言转义符： \
- 例如：\n表示换行、\\表示反斜杠、\'表示单引号、\"表示双引号、\t表示制表符（TAB）等。

```
>>>print("既需要'单引号'又需要\"双引号\"")
既需要'单引号'又需要"双引号"
```

字符串的索引

- 字符串是一个字符序列：字符串最左端位置标记为0，依次增加。对字符串中某个字符的检索被称为索引。索引的使用方式如下：

  <字符串或字符串变量>[序号]

- 如果字符串长度为L，正向递增需要以最左侧字符序号为0，向右依次递增，最右侧字符序号为L-1；反向递减序号以最右侧字符序号为-1，向左依次递减，最左侧字符序号为-L。

字符串的索引

- 字符串以Unicode编码存储，字符串的英文字符和中文字符都算作1个字符。

```
>>>"青青子衿,悠悠我心。"[-5]
'悠'
>>>s = "青青子衿,悠悠我心。"
>>>s[5]
'悠'
```

字符串的切片

- 对字符串中某个子串或区间的检索被称为切片。切片的使用方式如下：

<字符串或字符串变量>[N: M]

```
>>>"青青子衿,悠悠我心。"[8:4]
''
>>>"青青子衿,悠悠我心。"[:4]
'青青子衿'
>>>"青青子衿,悠悠我心。"[5:]
'悠悠我心。'
>>>print("青青子衿,悠悠我心。"[5:])
悠悠我心。
```

format()方法的基本使用

- 字符串format()方法的基本使用格式是：

<模板字符串>.format(<逗号分隔的参数>)

其中，模板字符串是一个由字符串和槽组成的字符串，用来控制字符串和变量的显示效果。槽用大括号({})表示，对应format()方法中逗号分隔的参数。

```
>>>"{}曰：学而时习之，不亦说乎。".format("孔子")
'孔子曰：学而时习之，不亦说乎。'
```

format()方法的基本使用

- 如果模板字符串有多个槽，且槽内没有指定序号，则按照槽出现的顺序分别对应.format()方法中的不同参数。

```
>>>"{}曰：学而时习之，不亦{}。".format("孔子","说乎")
'孔子曰：学而时习之，不亦说乎。'
```

format()方法的基本使用

- 可以通过format()参数的序号在模板字符串槽中指定参数的使用，参数从0开始编号

```
>>>"{1}曰：学而时习之，不亦{0}。".format("说乎","孔子")
'孔子曰：学而时习之，不亦说乎。'
```

```
>>>"{1}曰：{{学而时习之，不亦{0}}}。".format("说乎","孔子")"
'孔子曰：{学而时习之，不亦说乎}。'
```

format()方法的格式控制

- format()方法中模板字符串的槽除了包括参数序号，还可以包括格式控制信息。

  {<参数序号>:<格式控制标记>}

- 其中，格式控制标记用来控制参数显示时的格式。格式控制标记包括：<填充><对齐><宽度>,<.精度><类型>6个字段，这些字段都是可选的，可以组合使用

:	<填充>	<对齐>	<宽度>	,	<.精度>	<类型>
引导符号	用于填充的单个字符	< 左对齐 > 右对齐 ^ 居中对齐	槽的设定输出宽度	数字的千位分隔符 适用于整数和浮点数	浮点数小数部分的精度 或 字符串的最大输出长度	整数类型 b, c, d, o, x, X 浮点数类型 e, E, f, %

format()方法的格式控制

- <填充>、<对齐>和<宽度>主要用于对显示格式的规范。

- 宽度指当前槽的设定输出字符宽度，如果该槽参数实际值比宽度设定值大，则使用参数实际长度。如果该值的实际位数小于指定宽度，则按照对齐指定方式在宽度内对齐，默认以空格字符补充。
- 对齐字段分别使用<、>和^三个符号表示左对齐、右对齐和居中对齐。
- 填充字段可以修改默认填充字符，填充字符只能有一个。

format()方法的格式控制

```
>>>s = "等级考试"
>>>"{:25}".format(s)                    #左对齐，默认
'等级考试                                '
>>>"{:^25}".format(s)                   #居中对齐
'            等级考试            '
>>>"{:>25}".format(s)                   #右对齐
'                        等级考试'
>>>"{:*^25}".format(s)                  #居中对齐且填充*号
'**********等级考试**********'
>>>"{:+^25}".format(s)                  #居中对齐且填充+号
'++++++++++等级考试++++++++++'
>>>"{:十^25}".format(s)                 #居中对齐且填充汉字“十”
'十十十十十十十十十十等级考试十十十十十十十十十十'
>>>"{:^1}".format(s)                    #z指定宽度为1，不足变量s的宽度
'等级考试'
```

format()方法的格式控制

- <.精度><类型>主要用于对数值本身的规范
- <.精度>由小数点（.）开头。对于浮点数，精度表示小数部分输出的有效位数。对于字符串，精度表示输出的最大长度。小数点可以理解为对数值的有效截断。

```
>>>"{:.2f}".format(12345.67890)
'12345.68'
>>>"{:>25.3f}".format(12345.67890)
'                  12345.679'
>>>"{:.5}".format("全国计算机等级考试")
'全国计算机'
```

format()方法的格式控制

- <类型>表示输出整数和浮点数类型的格式规则。
- 对于整数类型，输出格式包括6种：

- b: 输出整数的二进制方式；
- c: 输出整数对应的Unicode字符；
- d: 输出整数的十进制方式；
- o: 输出整数的八进制方式；
- x: 输出整数的小写十六进制方式；
- X: 输出整数的大写十六进制方式；

```
>>>"{0:b},{0:c},{0:d},{0:o},{0:x},{0:X}".format(425)
'110101001,Σ,425,651,1a9,1A9'
```

format()方法的格式控制

- 对于浮点数类型，输出格式包括4种：

- e: 输出浮点数对应的小写字母e的指数形式；
- E: 输出浮点数对应的大写字母E的指数形式；
- f: 输出浮点数的标准浮点形式；
- %: 输出浮点数的百分形式。

```
>>>"{0:e},{0:E},{0:f},{0:%}".format(3.14)
'3.140000e+00,3.140000E+00,3.140000,314.000000%'
>>>"{0:.2e},{0:.2E},{0:.2f},{0:.2%}".format(3.14) # 对比输出
'3.14e+00,3.14E+00,3.14,314.00%'
```

format()方法的格式控制

- 常用的format()方法格式控制信息

```
>>>"{:.2f}".format(3.1415926)     # 输出小数点后两位
'3.14'
>>>"{:x}".format(1010)             # 输出整数的十六机制形式
'3f2'
>>>"{:.5}".format("这是一个很长的字符串")   # 输出字符串的前5位
'这是一个很'
>>>"{:-^10}".format("PYTHON")     # 居中并填充
'--PYTHON--'
```

字符串类型的操作

字符串操作符

- 针对字符串，Python语言提供了几个基本操作符

操作符	描述
x + y	连接两个字符串x与y
x * n 或 n * x	复制n次字符串x
x in s	如果x是s的子串，返回True，否则返回False

```
>>>name = "Python语言" + "程序设计"
>>>name
'Python语言程序设计'
>>>"等级考试!" * 3
'等级考试!等级考试!等级考试!'
>>>"语言" in name
True
>>>'Y' in name
False
```

字符串处理函数

- Python语言提供了一些对字符串处理的内置函数

函数	描述
len(x)	返回字符串x的长度，也可返回其他组合数据类型的元素个数
str(x)	返回任意类型x所对应的字符串形式
chr(x)	返回Unicode编码x对应的单字符
ord(x)	返回单字符x表示的Unicode编码
hex(x)	返回整数x对应十六进制数的小写形式字符串
oct(x)	返回整数x对应八进制数的小写形式字符串

```
>>>len("全国计算机等级考试Python语言科目")
19
>>>chr(10000)
'✏'
>>>hex(1010)
'0x3f2'
```

字符串处理方法

- 方法也是一个函数，只是调用方式不同。函数采用func(x)方式调用，而方法则采用**<a>.func(x)**形式调用。方法仅作用于前导对象<a>。

方法	描述
str.lower()	返回字符串str的副本，全部字符小写
str.upper()	返回字符串str的副本，全部字符大写
str.split(sep=None)	返回一个列表，由str根据sep被分割的部分构成
str.count(sub)	返回sub子串出现的次数
str.replace(old, new)	返回字符串str的副本，所有old子串被替换为new
str.center(width, fillchar)	字符串居中函数，fillchar参数可选
str.strip(chars)	从字符串str中去掉在其左侧和右侧chars中列出的字符
str.join(iter)	将iter变量的每一个元素后增加一个str字符串

字符串处理方法

- str.split(sep) 能够根据sep分隔字符串str，分割后的内容以列表类型返回。

```
>>>"Python is an excellent language.".split()
['Python', 'is', 'an', 'excellent', 'language.']
>>>"Python is an excellent language.".split('a')
['Python is ', 'n excellent l', 'ngu', 'ge.']
>>>"Python is an excellent language.".split('an')
['Python is ', ' excellent l', 'guage.']
```

- str.count(sub)方法返回字符串str中出现sub的次数，sub是一个字符串。

```
>>>"Python is an excellent language.".count('a')
3
```

字符串处理方法

- str.replace(old, new)方法将字符串str中出现的old字符串替换为new字符串，old和new的长度可以不同。

```
>>>"Python is an excellent language.".replace('a', '#')
'Python is #n excellent l#ngu#ge.'
>>>"Python is an excellent language.".replace('Python', 'C')
'C is an excellent language.'
```

- str.center(width, fillchar)方法返回长度为width的字符串，其中，str处于新字符串中心位置，两侧新增字符采用fillchar填充，当width小于字符串长度时，返回str。其中，fillchar是单个字符。

```
>>>"Python".center(20, "=")
'=======Python======='
>>>"Python".center(2, "=")
'Python'
```

字符串处理方法

- str.strip(chars)从字符串str中去掉在其左侧和右侧chars中列出的字符。chars是一个字符串，其中出现的每个字符都会被去掉。

```
>>>"   ==Python==   ".strip(' ')
'==Python=='
>>>"   ==Python==   ".strip(' =')
'Python'
>>>"   ==Python==   ".strip(' =n')
'Pytho'
```

- str.join(iter)中iter是一个具备迭代性质的变量，该方法将str字符串插入iter变量的元素之间，形成新的字符串。

```
>>>" ".join('PYTHON')
'P Y T H O N'
>>>",".join('12345')
'1,2,3,4,5'
>>>",".join(['1', '2', '3', '4', '5'])
'1,2,3,4,5'
```

类型判断和类型间转换

数字类型的转换

- Python语言提供**type(x)**函数对变量x进行类型判断，适用于任何数据类型。

```
n = eval(input('请输入一个数字:'))
if type(n) == type(123):
  print("输入的数字是整数。")
elif type(n) == type(11.3):
  print("输入的数字是浮点数。")
else:
  print("无法判断输入类型。")
```

数字类型的转换

- 数值运算操作符可以隐式地转换输出结果的数字类型，例如，两个整数采用运算符“/”的除法将可能输出浮点数结果。
- 此外，通过内置的数字类型转换函数可以显式地在数字类型之间进行转换

函数	描述
int(x)	将x转换为整数，x可以是浮点数或字符串
float(x)	将x转换为浮点数，x可以是整数或字符串
str(x)	将x转换为字符串，x可以是整数或浮点数

实例解析：恺撒密码

恺撒密码

- 恺撒撒密码是古罗马恺撒大帝用来对军事情报进行加密的算法，它采用了替换方法对信息中的每一个英文字符循环替换为字母表序列该字符后面第三个字符：

  原文：A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
  密文：D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

- 原文字符P，其密文字符C满足如下条件：

$$C = (P + 3) \bmod 26$$

- 解密方法反之，满足：

$$P = (C - 3) \bmod 26$$

恺撒密码-加密

- 恺撒密码的加密算法程序首先接收用户输入的文本，然后对字母a-z和字母A-Z按照密码算法进行转换

```
1  # CaesarEncode.py
2  ptxt = input("请输入明文文本: ")
3  for p in ptxt:
4      if "a" <= p <= "z":
5          print(chr(ord("a")+(ord(p)-ord("a")+3)%26), end='')
6      elif "A" <= p <= "Z":
7          print(chr(ord("A")+(ord(p)-ord("A")+3)%26), end='')
8      else:
9          print(p, end='')
```

```
>>>
请输入明文文本: This is an excellent Python book.
Wklv lv dq hafhoohqw Sbwkrq errn.
```

恺撒密码-解密

- 恺撒密码的解密算法程序首先接收用户输入的加密文本，然后对字母a-z和字母A-Z按照密码算法进行反向转换

```
1  # CaesarDecode.py
2  etxt = input("请输入加密后文本: ")
3  for p in etxt:
4      if "a" <= p <= "z":
5          print(chr(ord("a")+(ord(p)-ord("a")-3)%26), end='')
6     elif "A" <= p <= "Z":
7          print(chr(ord("A")+(ord(p)-ord("A")-3)%26), end='')
8     else:
9          print(p, end='')
```

```
>>>
请输入加密后文本: Wklv lv dq hafhoohqw Sbwkrq errn.
This is an excellent Python book.
```

本章小结

本章具体讲解了计算机中常用的数字类型及操作，包括Python数值运算操作符和数值运算函数。进一步讲解了字符串类型及格式化、字符串操作符、字符串处理函数和字符串处理方法等。最后，本章还介绍了类型判断和类型转换的基本方法，通过对恺撒密码及其变种若干实例的讲解帮助读者加深对数据类型操作的理解。

有没有一个人，你想给TA传个小纸条？用恺撒密码一展身手吧。