第3讲 Python Basics(1)

- 程序设计方法
- 程序语法元素
- 基本数据类型　　20190313
- 基本运算
- 字符串类型

Python优势

- 面向对象的解释型程序设计语言
- 强大的内置库和第三方库
- 语法简单灵活 20190313
- 开源、垮平台
- 注重如何解决问题

IPO程序编写方法

- 每个程序都有统一运算模式，即输入数据、处理数据和输出数据，形成程序的基本编写方法，即IPO（Input, Process, Output）
- 输入（Input）是一个程序的开始。
- 处理（Process）是程序对输入数据进行计算产生输出结果的过程。计算问题的处理方法统称为“算法”，它是程序最重要的组成部分
- 输出（Output）是程序展示运算结果的方式。

20190313

实例1 圆面积的计算

- IPO是程序设计的基本方法，也是描述计算问题的方式
- 例1: 圆面积的计算 20190313

  输入：圆半径r

  处理：计算圆面积area = π * r * r

  输出：圆面积area

```
In [1]: #例1：根据输入的圆半径计算圆面积

        #导入math库
        import math

        #输入半径
        r = float(input("请输入圆的半径 r: "))

        #计算面积
        s = math.pi * r ** 2

        #输出面积
        print("圆面积: ",s)
```

```
请输入圆的半径 r: 3.4
圆面积:  36.316811075498
```

计算机解决问题步骤

- 分析问题：分析问题的计算部分
- 划分边界：划分问题的功能边界
- 设计算法：设计问题的求解算法
- 编写程序：编写问题的计算程序
- 调试测试：调试和测试程序
- 升级维护：适应问题的升级维护

实例2 温度转换程序

- 温度体系：华氏温度与摄氏温度
- 问题： 如何利用Python进行两种温度转换
- 步骤1：分析问题的计算部分

20190313

采用公式转换方式解决计算问题

实例2 温度转换程序

- 步骤2：确定功能

  输入：华氏或者摄氏温度值、温度标识

  处理：温度转化算法 20190313

  输出：华氏或者摄氏温度值、温度标识

  F表示华氏度，82F表示华氏82度

  C表示摄氏度，28C表示摄氏28度

实例2 温度转换程序

- 步骤3：设计算法

  根据华氏和摄氏温度定义，转换公式：

  $C = (F - 32) / 1.8$ 20190313

  $F = C * 1.8 + 32$

  其中，C表示摄氏温度，F表示华氏温度

实例2 温度转换程序

- 步骤4：编写程序

```
#例2：温度转换程序

TemStr = input("请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)\n")
if TemStr[-1] in ['C', 'c']:
    f = float(TemStr[0:-1]) * 1.8 + 32
    print("转换后的维度为：%.2fF" %f)
elif TemStr[-1] in ['F', 'f']:
    c = (float(TemStr[0:-1])-32)/1.8
    print("转换后的维度为：%.2fC" %c)
else:
    print('输入错误！')
```

20190313

实例2 温度转换程序

- 步骤5：调试、运行程序

  在终端命令行运行命令：

  python TempConvert.py

  20190313

  或使用IDLE打开文件，按F5运行（推荐）

  输入数值，观察输出

```
#例2：温度转换程序

TemStr = input("请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)\n")
if TemStr[-1] in ['C', 'c']:
    f = float(TemStr[0:-1]) * 1.8 + 32
    print("转换后的维度为：%.2fF" %f)
elif TemStr[-1] in ['F', 'f']:
    c = (float(TemStr[0:-1])-32)/1.8
    print("转换后的维度为：%.2fC" %c)
else:
    print('输入错误！')
```

```
请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)
23f
转换后的维度为：-5.00C
```

语句缩进

- Python采用严格“缩进”表明程序的格式框架
- 缩进指每一行代码开始前的空白区域，用来表示代码之间的包含和层次关系
- 1个缩进 = 4个空格　　20190313

在Python中标明代码的层次关系唯一手段

```
TemStr = input("请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)\n")
if TemStr[-1] in ['C', 'c']:
    f = float(TemStr[0:-1]) * 1.8 + 32
    print("转换后的维度为：%.2fF" %f)
elif TemStr[-1] in ['F', 'f']:
    c = (float(val[0:-1])-32)/1.8
    print("转换后的维度为：%.2fC" %c)
else:
    print('输入错误！')
```

```
请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)
23C
转换后的维度为：73.40F
```

注释

- 注释：程序员在代码中加入的说明信息，不被计算机执行
- 单行注释用：#
- 多行注释用：’’’ ’’’或””” ”””

20190313

```
'''
温度刻画存在不同体系，摄氏度以1标准大气压下水的结冰点为0度，沸点为100度，
将温度进行等分刻画。华氏度以1标准大气压下水的结冰点为32度，沸点为212度，
华氏温度C与F摄氏温度之间的关系：C = (F-32)/1.8, F = C * 2.8 * 32
'''
#例2：温度转换程序

TemStr = input("请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)\n")
if TemStr[-1] in ['C', 'c']:
    f = float(TemStr[0:-1]) * 1.8 + 32
    print("转换后的维度为：%.2fF" %f)
elif TemStr[-1] in ['F', 'f']:
    c = (float(TemStr[0:-1])-32)/1.8
    print("转换后的维度为：%.2fC" %c)
else:
    print('输入错误！')
```

```
请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)
23f
转换后的维度为：-5.00C
```

多行显示

- 没有强制的语句终止符
- 一般以新行作为语句的结束符
- 使用斜杠（\）将一行代码分为多行显示

20190313

- 语句中包含 [], {} 或 () 括号就不需要使用多行连接符

```
In [1]: days = ('Monday', 'Tuesday', 'Wednesday',
                'Thursday', 'Friday', 'Saturday', 'Sunday')
```

input()函数

- input()用于在程序执行过程中接受用户输入内容，默认接受的输入内容为字符串类型
- input()函数可以包含一些提示性文字

<变量> = input(<提示性文字>)20190313

```
#例2：温度转换程序

TemStr = input("请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)\n")
if TemStr[-1] in ['C', 'c']:
    f = float(TemStr[0:-1]) * 1.8 + 32
    print("转换后的维度为：%.2fF" %f)
elif TemStr[-1] in ['F', 'f']:
    c = (float(TemStr[0:-1])-32)/1.8
    print("转换后的维度为：%.2fC" %c)
else:
    print('输入错误！')
```

请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)

```
#例2：温度转换程序

TemStr = input("请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)\n")
if TemStr[-1] in ['C', 'c']:
    f = float(TemStr[0:-1]) * 1.8 + 32
    print("转换后的维度为：%.2fF" %f)
elif TemStr[-1] in ['F', 'f']:
    c = (float(TemStr[0:-1])-32)/1.8
    print("转换后的维度为：%.2fC" %c)
else:
    print('输入错误！')
```

```
请输入温度及表示符号，如：23C(23摄氏度)，52F(52华氏度)
23f
转换后的维度为：-5.00C
```

print()函数

- print()用于在程序执行过程中输出信息
  - 直接输出字符信息

print 默认输出是换行的

```
In [2]: print("Hello")
        print("world!")

        Hello
        world!
```

20190313

如果要实现print输出不换行，需要指定结尾符 end=''

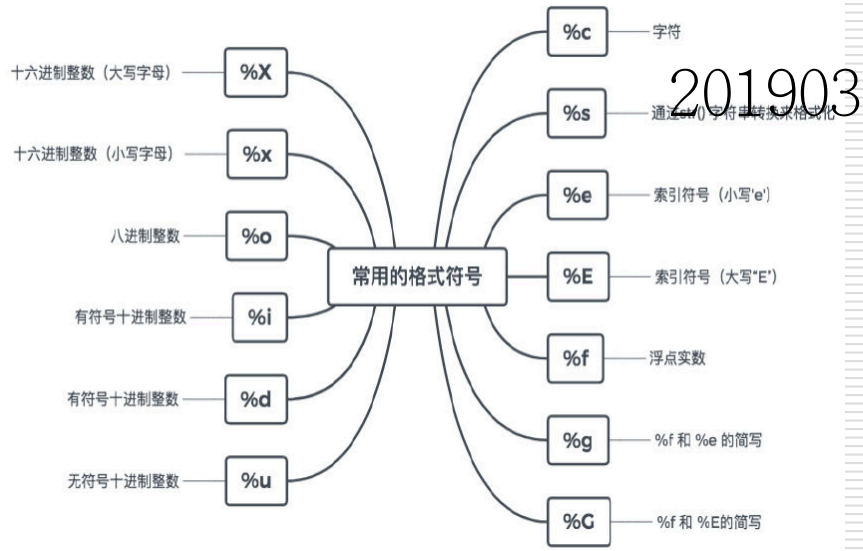
```
In [17]: print("Hello", end=' ')
         print("world!")

         Hello world!
```

  - 输出各种类型变量的值，并采用格式化输出方式

常用格式化符号

- 通过%选择要输出的变量
  - 格式化输出应用示例：print("你输入的名字是：%s"%name)



20190313

```
#多个变量同时输出，常使用格式化占位符输出
card_id = "234567"
pwd = 123

# print格式化输出
print("您输入的卡号是：%s"%card_id)
print("您输入的密码是：%s"%pwd)
```

```
您输入的卡号是：234567
您输入的密码是：123
```

```
#格式化输出浮点数，并指定精度
height = 180.35
print("您的身高是：%.2f"%height)
```

```
您的身高是：180.35
```

```
#格式化输出打印%，要使用%%表示是字符串而不是转换说明符
p = 99.99
print("您战胜了全国%.2f%%的用户"%p)
```

```
您战胜了全国99.99%的用户
```

```
# 利用*通配符从后面的元组中读取字段宽度或
import math
pi = math.pi
print(pi)
print("pi = %.*f" %(5,pi))
```

```
3.141592653589793
pi =  3.14159
```

常量与变量

- 常量：程序中值不发生改变的量
- 变量：程序中值可以发生改变的量
- 赋值：变量使用前须赋值，赋值号“=”
  - astring = “hello”　　20190313
  - 增量赋值：x += 1
  - 多重赋值：x = y = z = 1
  - 多元赋值：x, y, z = 1, 2, “hello”

```
r = float(input("请输入圆的半径 r: "))
#计算周长
p = 2 * math.pi * r
#计算面积
s = math.pi * r ** 2
#输出周长和面积
print("圆的周长: ",p)
print("圆的面积: ",s)
```

```
请输入圆的半径 r: 3.2
```

- 类型：Python中可将任意类型数据赋给变量，不需显式指定数据类型，解释器会根据具体赋值确定

类型、身份查看

- 任何类型值都是一个对象
- 函数：type(x)，返回x的类型，适用于所有类型的判断
- Python对象的三个特性
  - 身份：用id()查看
  - 类型：用type()查看
  - 值

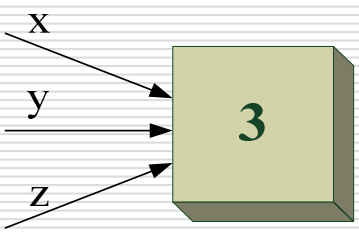
20190313

```
In [4]: a = 4.5
        b = 3
        print(id(a))
        print(id(b))
        print(type(a))
        print(type(b))

59142768
1545994832
<class 'float'>
<class 'int'>
```

变量引用

- 变量定义：在第一次赋值时自动声明
- Python是一种动态类型的语言
- 无处不在的引用：
  - Python中的每个值(而不是变量)都有一个唯一的标识
  - 任何变量名之间的赋值都是引用的赋值
    - x = 3
    - y = x
    - z = y




```
In [5]: #无处不在的引用
        x = 3
        y = x
        z = y
        print(id(x))
        print(id(y))
        print(id(z))

        print(type(x))
        print(type(y))
        print(type(z))
```

```
1545994800
1545994800
1545994800
<class 'int'>
<class 'int'>
<class 'int'>
```

标识符

- 标识符
  - 在python程序开发过程中，自定义的一些符号、名称
  - 由字母、数字、下划线（_）组成，不能以数字开头
  - 标识符区分大小写
- 命名规则
  - 见名知意，如：name
  - 驼峰命名法，如：类名（UserInfo）、异常名（ValueError）等
  - 小写字符+下划线，如：变量名（user_name）、函数名(get_name)
  - 不能够使用关键字
- 关键字
  - 在Python内部具有特殊功能的标识符
  - 通过keyword模块的kwlist函数查看

20190313

```
In [12]: #标识符区分大小写
         var_3=2
         Var_3=4.5
         print(var_3,Var_3)

2 4.5

In [13]: #错误标识符
         3_var

  File "<ipython-input-13-638c0cd91354>", line 2
    3_var
     ^

SyntaxError: invalid syntax
```

Python3直接支持中文符号，包括标识符名

```
In [53]: #Python3直接支持中文标识符

         变量1 = 8
         print(变量1)

8
```

关键字

- Python内部具有特殊功能的标识符，不能用作常数或变量，或任何其他标识符

and	elif	import	raise
as	else	in	return
assert	except	is	try
break	finally	lambda	while
class	for	nonlocal	with
continue	from	not	yield
def	global	or	True
del	if	pass	False
			None

Numbers数字型

- 数字类型用于存储数值，包括三种数字类型:
- int：默认为十进制，可以是2进制，8进制和16进制，没有取值范围限制
- float：带有小数点的数字，可用十进制和科学计数法表示。浮点数的数值范围及小数精度存在限制，与在不同计算机系统有关。
- complex：由实数部分和虚数部分构成，可用 z = a + bj 或 complex(a, b)表示，a是实部，b是虚部，a和b都是浮点类型，虚部分用j或者J标识

int整型

```
In [18]: #0b、0B代表2进制
         #0x、0X代表8进制
         #0X、0x代表16进制
         #多条语句可以放在一行，中间用分号“；”隔开
         var1 = 0b10; var2 = 0o10; var3 = 0x10
         print(var1,var2,var3)
         var1 = 0B10; var2 = 0O10; var3 = 0X10
         print(var1,var2,var3)
```

```
2 8 16
2 8 16
```

20190313

Python直接支持很长的整数

```
In [22]: var1 = 12345678910123456789012345678901

         print (var1, type(var1))
```

```
12345678910123456789012345678901 <class 'int'>
```

float浮点型

- 浮点型数值范围

```
In [19]: import sys
         sys.float_info

Out[19]: sys.float_info(max=1.7976931348623157e+308, max_exp=1024, max_10_exp=308, min=2.2250738585072014e-308, min_exp=-1021, min_10_exp=-307, dig=15, mant_dig=53, epsilon=2.220446049250313e-16, radix=2, rounds=1)
```

20190313

- 科学计数法：使用字母“e”或者“E”作为幂的符号，以10为基数，其形式：

  <a>e<b> = a * 10b

  例如，96e4, 6.7e15, 6.7e16, 9.6E5, -1.5

```
var1 = 96e4; var2 = 6.7e15; var3 = 6.7e16; var4 = 9.6E5; var5 = -1.5
print(var1,var2,var3,var4,var5)
```

```
960000.0 6700000000000000.0 6.7e+16 960000.0 -1.5
```

complex复数型

- 示例

z = 1.23e-4 + 5.6e+89j（实部和虚部是什么？）

对于复数z，用z.real获取实部，z.imag获取虚部

即：z.real=0.000123，z.imag=5.6e+89

```
z = 1.23e-4 + 5.6e+89j
print(z.real, z.imag)
```

```
0.000123 5.6e+89
```

复数由实数部分和虚数部分构成，可以用 a + bj,或者 complex(a,b) 表示，复数的实部 a 和虚部 b 都是浮点型

```
In [24]: var1 = 3+5.3j; var2 = complex(3.4e5,7.8)

         print (var1, type(var1), var2, type(var2))
```

```
(3+5.3j) <class 'complex'> (340000+7.8j) <class 'complex'>
```

数值运算

- 三种类型存在一种逐渐“扩展”的关系：

  int -> float-> complex

  （整数是浮点数特例，浮点数是复数特例）

- 不同数字类型可混合运算，运算后生成结果是“更宽”的类型

  123 + 4.0 = 127.0

  int + float = float

```
In  [23]: 123 + 4.0

Out[23]: 127.0
```

数值转换

- 数值运算操作符可隐式转换输出结果的数字类型
- 例如，两个整数采用运算符“/”除法，可能输出浮点数结果

此外，通过内置数字类型转换函数可显式进行数字类型转换

20190313

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
complex(re[, im])	生成一个复数，实部为re，虚部为im，re可以是整数、浮点数或字符串，im可以是整数或浮点数但不能为字符串

```
In [27]: #数据类型转化
         print(int(4.5))
         print(float(4))
         print(complex(4))

         4
         4.0
         (4+0j)
```

数值函数

- Python提供与数值运算相关的内置函数

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

20190313

```
# python内置的常用数值运算函数
print(abs(-11))
print(divmod(4,3))
print(pow(2,3))
print(round(4.55,1))
print(max(5,9,3))
print(min(5,9,3))
```

```
11
(1, 1)
8
4.5
9
3
```

- 问题如何查看Python的内置函数？

Boolean布尔型

- bool值：True and False

布尔类型转换值	数值型	非数值型
False	整数 0	None
	浮点数 0.0	空字符串''或"""
		空集合(),[]或{}
True	其余	其余

布尔类型, bool 值: True and False

20190313

```
In [25]: i_love_you = True
         you_love_me = False
         print(i_love_you, type(i_love_you))
         print(you_love_me, type(you_love_me))

         True <class 'bool'>
         False <class 'bool'>
```

算术运算

算术运算符	功能描述	实例（设 x = 10，y = 3）
+	加：两个数字相加	x + y = 13
-	减：两个数字相减	x - y = 7
*	乘：两个数字相乘	x * y = 3.3333333333333335
/	除：x 除 y	x / y = 13
%	取模：返回除法的余数	x % y = 1
**	幂：返回 x 的 y 次幂	x ** y = 1000
//	取整：返回商的整数部分（向下取整）	x // y = 3.0


```
#算术运算操作
print(100 / 3)
print(100 // 3)
print(4 % 3)
print(2 ** 3)
```

```
33.333333333333336
33
1
8
```

赋值运算

赋值运算符	功能描述	实例（设 x = 10，y = 3）
=	简单赋值运算符	z = x + y，将 x+ y 的结果赋给 z
+=	加法赋值运算符	z += x 等效于 z = z + x
-=	减法赋值运算符	z -= x 等效于 z = z - x
*=	乘法赋值运算符	z *= x 等效于 z = z * x
/=	除法赋值运算符	z /= x 等效于 z = z / x
%=	取模赋值运算符	z %= x 等效于 z = z % x
**=	幂赋值运算符	z **= x 等效于 z = z ** x
//=	取整赋值运算符	z //= x 等效于 z = z // x


```
#增量赋值
a = 10
a = a + 5
print(a)
a *= 5
print(a)
```

```
15
75
```

比较运算

比较运算符	功能描述	实例（设 x = 10，y = 3）
==	等于	x == y 返回 False
!=	不等于	x != y 返回 True
>	大于	x > y 返回 False
<	小于	x < y 返回 False
>=	大于等于	x >= y 返回 False
<=	小于等于	x <= y 返回 False

20190313

逻辑运算

逻辑运算符	功能描述	实例（设 x=1，y=0）
and	逻辑“与”：如果 x 为 False，x and y 返回 x，否则返回 y	x and y 返回 0
or	逻辑“或”：如果 x 为非 0，返回 x 的值，否则返回 y	x or y 返回 1
not	逻辑“非”：如果 x 为 True，返回 False，否则返回 y	nor x 返回 False

注：在逻辑运算中，True 的值就是 1，False 的值就是 0

运算优先级

表 1 运算符的优先级

运算符说明	Python运算符	优先级
索引运算符	x[index]或x[index:index2[:index3]]	18、19
属性访问	x.attrbute	17
乘方	**	16
按位取反	~	15
符号运算符	+或-	14
乘、除	*、/、//、%	13
加、减	+、-	12
位移	>>、<<	11
按位与	&	10
按位异或	^	9
按位或	|	8
比较运算符	==、!=、>、>=、<、<=	7
is运算符	is、is not	6
in运算符	in、not in	5
逻辑非	not	4
逻辑与	and	3
逻辑或	or	2

```
# 运算的优先级
a = 40 - 3 ** 2 + 11 / 3 ** 2 * 8
print(a)
```

```
40.77777777777778
```

20190313

运算优先级

1、将下列数据表达式用python程序写出，并运算结果

（1） a = (24+7-3x4)/5

（2）b = (1+32)x(16mod7)/7

（3）假设c=1, c*=3+5**的运算结果？

2、思考各种操作符的优先级，计算下列表达式

（1）30-3**2+8//3**2*10

（2）3*4**2/8%5

（3）2**2**3

（4）(2.5+1.25j)*4j/2

String字符串

- 字符串是字符的序列，可以由单引号‘’，双引号“”或三引号’’’ ’’’构成

- 字符串类型变量定义

  s = "hello" 或者'hello'

- 组成字符串的方式
  - 使用“+”号将两个字符串连接成一个新的字符串
  - 使用字符串格式化符号
- 下标

h	e	l	l	o
0	1	2	3	4

  - 通过下标获取指定位置的字符：string_name[index]
- 切片
  - 切片的语法：string_name[起始:结束:步长]

20190313

```
In [36]: '''
字符串类型
'''
print('单引号表示可以使用"双引号"作为字符串的一部分')
print("双引号表示可以使用‘单双引’作为字符串的一部分")
print('''三引号中可以使用“双引号”‘单双引’，
也可以使用换行''')
```

```
单引号表示可以使用"双引号"作为字符串的一部分
双引号表示可以使用‘单双引’作为字符串的一部分
三引号中可以使用“双引号”‘单双引’，
也可以使用换行
```

String字符串

- 字符串的两种索引：正向索引和逆向索引：


```
In [35]: #字符串存在两种索引：正向索引和逆向索引
         s = "hello"
         print(s[4])
         print(s[-1])
         #区间访问方式
         print(s[0:6])

o
o
hello
```



```
In [40]: #字符串以Unicode编码存储，英文字符和中文字符都算作一个字符
         name = "Python语言程序设计"
         print(name[0])
         print(name[7])
         print(name[6:8])

P
言
语言
```

String字符串

- 反斜杠（\）在字符串中表示转义，即该字符与后面相邻的一个字符共同组成了新的含义，
- 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

20190313

```
In [42]: '''
         反斜杠（\）在字符串中表示转义，即该字符与后面相邻的一个字符共同组成了新的含义，
         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("Python\n语言\t程序")

Python
语言    程序
```

字符串基本操作符

- x + y：连接两个字符串
- x*n or x*n：复制n次字符串x
- x in s：x是否是s的子串 20190313
- str[i]：索引，返回第i个字符
- str[*n*:m]：切片，返回指定范围的字符串

```
In [44]: #基本的字符串操作
         name = "Python"
         v = "3.x"
         print(name + v)
         print(name * 3)
         print(3 * name)
         print("p" in name)
         print("P" in name)
         print(name[5])
         print(name[3:8])
```

```
Python3.x
PythonPythonPython
PythonPythonPython
False
True
n
hon
```

字符串处理函数

- len(x)：获取子串长度
- str(x)：转成子串型
- chr(x)：返回Unicode编码对应的单字符
- ord(x)：返回单字符对应的Unicode编码
- hex(x)：将10进制整数转换成16进制字符串
- oct(x)：将整数转换成8进制字符串

20190313

```
In [45]: #字符串处理函数
         print(len("python语言程序设计"))
         print(str(3.1415926))
         print(hex(255))
         print(oct(-255))
```

```
12
3.1415926
0xff
-0o377
```

字符串常用内置函数

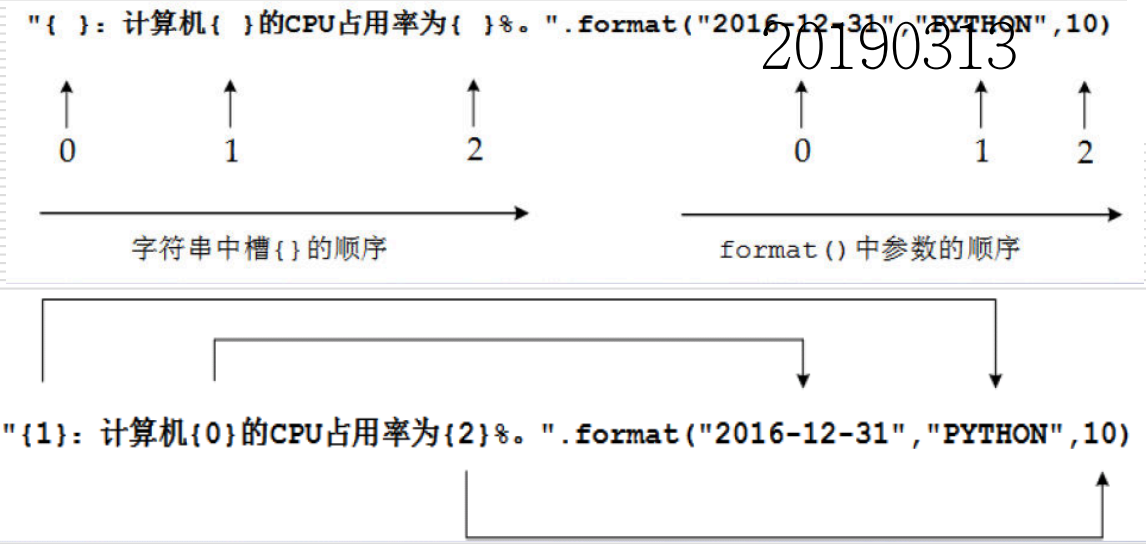
方法	描述
str.lower()	返回字符串str的副本，全部字符小写
str.upper()	返回字符串str的副本，全部字符大写
str.islower()	当str所有字符都是小写时，返回True，否则False
str.isprintable()	当str所有字符都是可打印的，返回True，否则False
str. isnumeric()	当str所有字符都是字符时，返回True，否则False
str.isspace()	当str所有字符都是空格，返回True，否则False
str.endswith(suffix[,start[,end]])	str[start: end] 以suffix结尾返回True，否则返回False
str.startswith(prefix[, start[, end]])	str[start: end] 以suffix开始返回True，否则返回False
str.split(sep=None, maxsplit=-1)	返回一个列表，由str根据sep被分割的部分构成
str.count(sub[,start[,end]])	返回str[start: end]中sub子串出现的次数
str.replace(old, new[, count])	返回字符串str的副本，所有old子串被替换为new，如果count给出，则前count次old出现被替换
str.center(width[, fillchar])	字符串居中函数，详见函数定义
str.strip([chars])	返回字符串str的副本，在其左侧和右侧去掉chars中列出的字符
str.zfill(width)	返回字符串str的副本，长度为width，不足部分在左侧添0
str.format()	返回字符串str的一种排版格式，3.6节将详细介绍
str.join(iterable)	返回一个新字符串，由组合数据类型（见第6章）iterable变量的每个元素组成，元素间用str分割

20190313

字符串类型格式化

- Print() 函数用槽格式和format()方法将变量和字符串结合到一起输出
- format()方法基本使用格式：

  <模板字符串>.format(<逗号分隔的参数>)




```
card_id = "234567"
pwd = 123
print("您输入的卡号是：{}，您输入的密码是：{}".format(card_id,pwd))
height = 180.35
print("您的身高是：{:.2f}".format(height))
```

```
您输入的卡号是：234567，您输入的密码是：123
您的身高是：180.35
```

```
# 拼接字符串：接受输入的两个字符串，将它们组合后输出
str1 = input("请输入一个人的名字：")
str2 = input("请输入一个国家名字：")
print("世界这么大，{}想去{}。".format(str1,str2))
```

```
请输入一个人的名字：我
请输入一个国家名字：中国
世界这么大，我想去中国。
```

字符串类型格式化

- 模板字符串的槽除了包括参数序号，还可包括格式控制信息
- 槽的内部样式：{<参数序号>: <格式控制标记>}

格式控制标记用来控制参数显示时的格式，包括 6个字段:

<填充>、<对齐>、<宽度>、<.精度>、<类型>

这些字段都是可选的，可以组合使用

20190313

:	<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, %

类型转换

```
#类型转换
name = input("请输入姓名：")
age = input("请输入年龄")
print("name: %s,age:%d"%(name,int(age)))
print(int("123"))
a = int("123") #字符串转整型
b = float("3.14")#字符串转浮点型
c = str(345)
print(type(a))
print(type(b))
print(type(c))
```

```
请输入姓名：wangyi
请输入年龄20
name: wangyi,age:20
123
<class 'int'>
<class 'float'>
<class 'str'>
```

```
#eval(str)把字符串自动转换成合适的数据类型
a1= eval("123")
a2 = eval("3.14")
print(type(a1))
print(type(a2))
```

```
<class 'int'>
<class 'float'>
```

20190313

实例3 输出对应月份名称缩写

- 输入一个月份数字，返回对应月份名称缩写

  这个问题的IPO模式：

  输入：输入一个表示月份的数字(1-12)

  处理：利用字符串基本操作实现该功能

  输出：输入数字对应月份名称的缩写

实例3 输出对应月份名称缩写

- 将所有月份名称缩写存储在字符串中
- 在字符串中截取适当的子串来查找特定月份
- 找出在哪里切割子串　　20190313
- 每个月份的缩写都由3个字母组成，如果pos表示一个月份的第一个字母，则months[pos:pos+3]表示这个月份的缩写，即：

  monthAbbrev = months[pos:pos+3]

实例3 输出对应月份名称缩写

In [46]:
```
#输入一个月份数字，返回对应月份英文名称缩写
#month.py

months = "JanFebMarAprMayJunJulAugSepOctNovDec"
n = input("请输入月份数（1-12）：")
pos = (int(n) - 1) * 3
monthAbbrev = months[pos:pos+3]
print("月份简写：" + monthAbbrev + ".")
```

```
请输入月份数（1-12）：3
月份简写：Mar.
```

实例4 凯撒密码

- 假设用户输入仅包含小写字母a~z和空格，请编写一个程序， 对输入字符串进行凯撒密码加密，直接输出结果，其中空格不用加密， 使用input()获得输入，例如，输入: python is good，输出: sbwkrq lv jrrg

20190313

```
In [4]: # 例4：恺撒密码

Str = input('请输入名文：')
for i in range(0, len(Str)):
    if Str[i] == ' ':
        print(' ', end="")
    elif Str[i] in ['x', 'y', 'z']:
        print(chr(ord(Str[i])-23),end='')
    else:
        print(chr(ord(Str[i])+3),end='')
```

```
请输入名文：a
d
```

学习编程的武功秘籍

- 首先，掌握编程语言的语法，熟悉基本概念和逻辑
- 其次，结合计算问题思考程序结构，会使用编程套路
- 最后，参照案例多练习多实践，学会举一反三次

20190313

源自： 嵩天, 礼欣, 黄天羽, 著. Python语言程序设计(第2版). 北京: 高等教育出版社, 2017.2