Python3 基础语法

# 编码

默认情况下，Python 3 源码文件以 UTF-8 编码

# 缩进规范

## 行缩进

python最具特色的就是**使用缩进来表示代码块**，不需要使用大括号 {} 。

缩进的空格数是可变的，但是同一个代码块的语句必须包含相同的缩进空格数。实例如下：

In [63]:

```
if True:
    print ("True")
else:
    print ("False")
```

True

## 空行

函数之间或类的方法之间用空行分隔，便于日后代码的维护或重构，表示一段新的代码的开始。

# 多个语句构成代码组

缩进相同的一组语句构成一个代码块，我们称之代码组。

像if、while、def 和 class 这样的复合语句，首行以关键字开始，以冒号( : )结束，该行之后的一行或多行代码构成代码组。

我们将首行及后面的代码组称为一个子句(clause)。

# import 与 from...import

在 python 用 import 或者 from...import 来导入相应的模块。

1. 将整个模块(somemodule)导入，格式为： import somemodule
2. 从某个模块中导入某个函数,格式为： from somemodule import somefunction
3. 从某个模块中导入多个函数,格式为： from somemodule import firstfunc, secondfunc, thirdfunc
4. 将某个模块中的全部函数导入，格式为： from somemodule import *

# 注释

1. **单行注释** 以 # 开头
2. **多行注释** 用三个单引号 ''' 或者三个双引号 """ 将注释括起来

In [64]:

```
# 这是一个注释
print("Hello, World!")


'''
这是多行注释，用三个单引号
这是多行注释，用三个单引号
这是多行注释，用三个单引号
'''
print("Hello, World!")


"""
第五注释
第六注释
"""
print ("Hello, Python!")
```

```
Hello, World!
Hello, World!
Hello, Python!
```

# 标识符

1. **第一个字符**必须是字母表中**字母或下划线 _**。
2. 标识符的其他的部分由字母、数字和下划线组成。
3. 标识符对**大小写敏感。**

# Python3 运算符

## 赋值运算符

1. = 简单的赋值运算符
2. 混合运算赋值运算符
    A. c += a 等效于 c = c + a 以此类推

# 算术运算符

In [65]:

```
5 + 4  # 加法
```

Out[65]:

9

In [66]:

```
4.3 - 2 # 减法
```

Out[66]:

2.3

In [67]:

```
3 * 7  # 乘法
```

Out[67]:

21

In [68]:

```
2 / 4  # 除法，得到一个浮点数
```

Out[68]:

0.5

In [69]:

```
2 ** 5 # 乘方
```

Out[69]:

32

In [70]:

```
17 % 3 # 取余（只允许整型）
```

Out[70]:

2

In [71]:

```
2 // 4 # 除法，得到一个整数
```

Out[71]:

0

In [72]:

```
7.0 // 2 # //得到的并不一定是整数类型的数，它与分母分子的数据类型有关系
```

Out[72]:

3.0

## 比较运算符

In [73]:

```
1 == 1 # 等于
```

Out[73]:

True

In [74]:

```
1 != 2 # 不等于
```

Out[74]:

True

In [75]:

```
1 < 2 # 小于
```

Out[75]:

True

In [76]:

```
1 > 0 # 大于
```

Out[76]:

True

In [77]:

```
1 <= 1 # 小于等于
```

Out[77]:

True

In [78]:

```
1 >= 1 # 大于等于
```

Out[78]:

True

## 逻辑运算符

1. and 且
2. or 或
3. not 非

## 成员运算符

1. in 如果在指定的序列中找到值返回 True，否则返回 False。
2. not in

In [79]:

```
a = 1
b = 6
list = [1, 2, 3, 4, 5 ];

if ( a in list ):
   print ("1 - 变量 a 在给定的列表中 list 中")

if ( b not in list ):
   print ("2 - 变量 b 不在给定的列表中 list 中")
```

```
1 - 变量 a 在给定的列表中 list 中
2 - 变量 b 不在给定的列表中 list 中
```

## 身份运算符

1. is 判断两个标识符是不是引用自一个对象(存储单元), x is y, 类似 id(x) == id(y).
2. is not

In [80]:

```
a = 20
b = 20
c = b
d = 30

if ( a is b ):
   print ("1 - a 和 b 有相同的标识")
else:
   print ("1 - a 和 b 没有相同的标识")

if ( c is b ):
   print ("2 - c 和 b 有相同的标识")
else:
   print ("2 - c 和 b 没有相同的标识")

if ( d is b ):
   print ("3 - d 和 b 有相同的标识")
else:
   print ("3 - d 和 b 没有相同的标识")
```

```
1 - a 和 b 有相同的标识
2 - c 和 b 有相同的标识
3 - d 和 b 没有相同的标识
```

# 位运算符

按位运算符是把数字看作二进制来进行计算的

1. & 与， | 或， ^ 异或， ~ 非
2. << m : 左移动运算符：把"<<"左边的运算数的各二进位全部左移m位，空者补0
3. >> m : 右移动运算符：把">>"左边的运算数的各二进位全部右移m位，空者补0

In [81]:

```
a = 60            # 60 = 0011 1100
b = 13            # 13 = 0000 1101
c = 0

c = a & b;        # 12 = 0000 1100
print ("1 - c 的值为：", c)

c = a | b;        # 61 = 0011 1101
print ("2 - c 的值为：", c)

c = a ^ b;        # 49 = 0011 0001
print ("3 - c 的值为：", c)

c = ~a;           # -61 = 1100 0011
print ("4 - c 的值为：", c)

c = a << 2;       # 240 = 1111 0000
print ("5 - c 的值为：", c)

c = a >> 2;       # 15 = 0000 1111
print ("6 - c 的值为：", c)
```

```
1 - c 的值为： 12
2 - c 的值为： 61
3 - c 的值为： 49
4 - c 的值为： -61
5 - c 的值为： 240
6 - c 的值为： 15
```

## 运算符优先级

以下表格列出了从最高到最低优先级的所有运算符：

| 运算符 | 描述 |
| --- | --- |
| ** | 指数 (最高优先级) |
| ~ + - | 按位翻转, 一元加号和减号 (最后两个的方法名为 +@ 和 -@) |
| * / % // | 乘，除，取模和取整除 |
| + - | 加法减法 |
| >> << | 右移，左移运算符 |
| & | 位 'AND' |
| ^ \| | 位运算符 |
| <= < > >= | 比较运算符 |
| <> == != | 等于运算符 |
| = %= /= //= -= += *= **= | 赋值运算符 |
| is is not | 身份运算符 |
| in not in | 成员运算符 |
| and or not | 逻辑运算符 |

# Print 输出

## 输出换行

1. print 默认输出是换行的
2. 不换行需要在变量末尾加上 end=""

In [82]:

```
x="a"
y="b"
# 换行输出
print( x )
print( y )

print('---------')
# 不换行输出
print( x, end=" " )
print( y, end=" " )
print()
```

```
a
b
---------
a b
```

## 输出字符串和数字

In [83]:

```
print("runoob")      # 输出字符串
```

```
runoob
```

In [84]:

```
print(100)           # 输出数字
```

```
100
```

In [85]:

```
str = 'runoob'
print(str)           # 输出变量
```

```
runoob
```

In [86]:

```
L = [1,2,'a']        # 输出列表
print(L)
```

```
[1, 2, 'a']
```

In [87]:

```
t = (1,2,'a')        # 输出元组
print(t)
```

(1, 2, 'a')

In [88]:

```
d = {'a':1, 'b':2}   # 输出字典
print(d)
```

{'a': 1, 'b': 2}

# 格式化输出整数

支持参数格式化，与 C 语言的 printf 类似

- print("%s,%d" %('runoob',2))

In [89]:

```
str = "the length of (%s) is %d" %('runoob',len('runoob'))
print(str)
```

the length of (runoob) is 6

python字符串格式化符号

1. %c : 格式化字符及其ASCII码
2. %s : 格式化字符串
3. %d : 格式化整数
4. %f : 格式化浮点数字，可指定小数点后的精度
5. %e : 用科学计数法格式化浮点数

# 格式化输出浮点数(float)

In [94]:

```
pi = 3.141592653
print('%10.3f' % pi)          #字段宽10，精度3
```

     3.142

In [91]:

```
pi = 3.141592653
print('%010.3f' % pi)         #字段宽10，精度3,用0填充空白
```

000003.142

In [92]:

```
pi = 3.141592653
print('%-10.3f' % pi)          #左对齐
```

3.142

In [93]:

```
pi = 3.141592653
print('%+f' % pi)              #显示正负号
```

+3.141593