day03笔记：布尔运算，字符串

笔记本： Python基础
创建时间： 2018/6/30 17:11 **更新时间：** 2018/7/2 9:40
作者： liuchang_0412@163.com

1.布尔运算:

运算符:

not and or

①not 运算符作用:

逻辑取反

语法:

not 表达式

例:

```
not True     # False
not False    # True
not 100      # False  相当于 not bool(100)
not 0.0      # True
not 0J       # True
not 1+2J     # False
not ''       # True
not "False"  # False
not None     # True
```

#"False"(字符串) 和 False(布尔类型)

②and 运算符:

作用:

返回优先为假值(False)的对象

语法:

表达式1 and 表达式2

示例:

```
0 and 100   # 返回0
100 and 200 # 返回200
100 and 0   # 返回0
```

说明:

当表达式1的布尔测试值为True时，返回 表达式2,
否则返回表达式１

bool(表达式1) and bool(表达式2)　结果的布尔测试值:

False	False	False
False	True	False
True	False	False
True	True	True

③or 运算符:
作用:
优先返回值为真(True)的对象
语法:
表达式1 or 表达式2
示例:
0 or 100 返回值是100
100 or 0 返回值 100
0.0 or 0 返回 0
3.14 or 0.618 返回 3.14

④正负号运算符 +(正号) -(负号)
语法:
+ 表达式
- 表达式
说明:
一元运算符

⑤位运算运算符:
#bin(x)函数: bin(x) 将x转换为二进制(binary)的字符串
& 位与
| 位或
^ 位异或
<< 左移
>> 右移
~ 求反

& 位与:
语法
表达式x & 表达式y
作用:
按位操作,两个对应的位都为1,则结果为1
两个对应的位只要有一个为0,则结果为0

| 位或
语法:

表达式x | 表达式y
作用:
按位操作,两个对应的位只要有一个为1,则结果为1
两个对应的位都为0，则结果为0

^ 位异或
语法:
表达式x ^ 表达式y
作用:
按位操作,两个对应的位不同，结果为1
两个对应的位相同，结果为0

<< 左移运算
语法格式:
表达式x << 整数表达式y
作用:
将x的二进制值，按位向左移动y位，低位补0
示例:
x = 6 << 1

>> 右移运算
语法格式:
表达式x >> 整数表达式y
作用:
将x的二进制值,按位向右移动y位，低位溢出丢弃
示例:
6 >> 1 # 3
6 >> 2 # 1
6 >> 3 # 0
6 >> 100 # 0

~ 按位求反
语法格式:
~ 表达式
作用:
将数据二进制相应位取反
示例:
~ 6

#练习:
IP = 0xc0a80164 # IP: 192.168.1.100

MASK = 0xFFFFFF00 # 子网俺码 255.255.255.0
求：IP & MASK 的值 # 网段地址:192.168.1.0
求: IP | ~MASK 的值 # 广播地址:192.168.1.255

#hex(x) 将一个整数转为16进制的字符串
#255 ->0b11111111

2.字符串 str
作用是用来记录文本信息
字符串是一个有序的字符序列

如何表示一个字符串:
在非注释中凡是用引号括起来的部分都是字符串

' 单引号
" 双引号
''' 三单引号
""" 三双引号

空字符串的表示方法:

''
""
''''''
""""""

空字符的的布尔值(bool)为False.

非空字符中表示方法:
'hello'
"hello"
"""hello"""
'''hello'''
单引和双引号三引号区别:
单引号内的双引号不算结束符
双引号内的单引号不算结束符

I'm a teacher
"I'm a teacher"

I am "weimingze"

```
    print('I am "weimingze"')
```
换行符:
```
    \n
    print('hello\nworld!')
    print("""hello
    world!""")
```

三引号字符串，也叫所见即所得字符串,字符串的换行直接被记录在字符串内部

隐式字符串拼接:

```
x = "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"
    "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"
    "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"
    "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"
s = "I'm " 'a teach.' 'name:"weimz"'
```

用转义序列代表特殊字符
字符串常量中用字符反斜杠 \ 后跟一些字符代表特定含义的字符
```
    \'   单引号
    \"   双引号
    \n   换行符
```

字符串中反斜杠字符表
```
    \'
    \"
    \n          换行符
    \a          响铃
    \\          一个反斜杠字符 \
    \r          返回光标至首行
    \f          换页字符
    \t          水平制表符
    \v          垂直制表符
    \b          退格(backspace)
    \0          字符零
    \xXX        十六进制表示
    \uXXXX      Unicode16位的十六进制表示
    \uXXXXXXXX Unicode32位的十六进制表示
```

ASCII字符表
```
    $ man ascii<回车>
```

中文 对应 "\u4e2d\u6587"
    Unicode
    UTF-8 （Unicode Transfrom Font 8）

常用ASCII编码:

字符	十进制	十六进制
'0'	48	0x30
'A'	65	0x41
'a'	97	0x61

raw字符串(原始字符串)
    格式:
        r'C:\Windows\System32'
        'C:\\Windows\\System32'
  作用:
      将字符串内的反斜杠不做为转义字符
  示例:
      a='C:\newfile\test.py'
      print(a)
      a=r'C:\newfile\test.py'
      print(a)

序列相关的字符串函数:
    len(seq) 函数用于获取字符串序列的长度length
        (长度是指字节数)
    len("ABCD") # 4
    max(seq) 函数用于获取一个字符串是最大值的字符
    max("Aa1Bb2")   # b
    min(seq) 函数用于获取一个字符串是最小值的字符
    min("Aa1Bb2")   # 1
字符串代码转换函数:
    ord(str) 返回一个字符的ASCII的值
    chr(i)   返回i这个值所对应的字符

数字转换为字符串函数
    hex(i) 将整数转换为十六进制字符串
    oct(i) 将整数转换为八进制字符串
    bin(i) 将整数转换为二进制字符串

将字符串转换为整数(之前以讲过)
    int(x, base=0) 将字符串转换为整数
    float
    bool
    complex

将对象转换为字符串类型str函数
    str(x)   x可以为任意对象

为什么要字符串转换?
    "123" + 4
    它的含意可能是:
    123+4   -> 127
    int("123") + 4 -> 127
    还可能是
    "123" + "4" -> "1234"
    "123" + str(4) -> "1234"

字符串的运算:
  + += * *=
  + 加号运算符用于拼接字符串
      例:
      a = "ABCD"
      b = "EFG"
      c = a + b # c---> "ABCDEFG"
  += 运算符用原字符串与右侧字符串拼接生成新的字符串
     例:
       a = "ABCD"
       a += "EFG"   # a ---> "ABCDEFG"
  * 运算符生成重复字符串
    例:
    x = "ABCD" * 3
    注:
      *号右侧的对象只能是整数

  *= 运算符生成重复字符串并改变变量的绑定
   x = "123"
   x *= 4   # 等同于 x = x * 4
   print(x)
    注:

*= 右侧的对象只能是整数

in / not in 运算符
语法:
x in s
作用:
in/not 用于序列，字典，集合中，
用于判断某个值是否存在于对象中,
存在返回True,不存在返回False
注:
not in 与 in返回值相反

例:
```
s = "welcome to tarena!"
if 'to' in s:
    print("to在字符串s中")
else:
    print("to不在字符串s中")
```

字符串比较:
>
>=
<
<=
==
!=
例:
```
"ABC" > "ABB"  # True
"ADC" < "ABC"  # False
"ABC" >= "123" # True
"AD"  >= "ABC" # True
"AB"  < "ABC"  # True
"ABC" == "abc" # False
"ABCD" != "DCBA"  # True
```

Python 运算符优先级:
(自上而下，由高到低)
** 指数
~, +, - 按位反转,一元正号，一元负号
*, /, //, % 乘....

+, -　　加法减法
>>, <<　　右移，左移
&　　位与
^　　位异或
|　　位或
< <= > >= ==　　!=, is, is not, in, not in 比较等..
not　　布尔非(取非)
and
or
if - else　　表件表达式

练习1：
用字符串 * 运算符打印三角形
要求输入一个整数，此整数代表此三角形离左侧的字符数
$ python3 tri_angle.py
请输入离左侧的距离: 3
```
   *
  ***
 *****
*******
```

练习2：
输入三行文字 ,让这三行文字在一个方框内居中显示
如输入(不要输入中文):
hello tarena!
my name is weimingze!
good-bye
显示结果如下：
```
+----------------------+
|     hello tarena!    |
| my name is weimingze! |
|       good-bye       |
+----------------------+
```