字符串

字符串作为编程中一个运用很多的类型，我们就多说一点吧

字符串是 Python 中最常用的数据类型。我们可以使用引号(‘或”)来创建字符串
创建字符串很简单，只要为变量分配一个值即可。例如：

```
var1 = ‘Hello World!’
var2 = "Python"
```

- 字符串中的单个值可以用[]来访问

var1[0]的值为H

- 字符串可以用+拼接

```
var3 = var1 + var2
```

此时var3的值为“Hello World!Python”

字符串的切片

`str = 'Hello!'`

索引(0, 1, 2, 3, 4, 5)或(–5, –4, –3, –2, –1)

`arr[i:j]`

i为起始值start，j为终止值end

[start,end)为前闭后开的区间

i缺省值为0，j缺省值为len(str) – 1

`str[1:4]`的值为`ell`

`str[:4]`的值为`Hell`

字符串中是否有某个字符

str = ‘Hello!’

可以用in和not in判断

"H" in str的值为True

"M" not in str的值为True

成员运算符

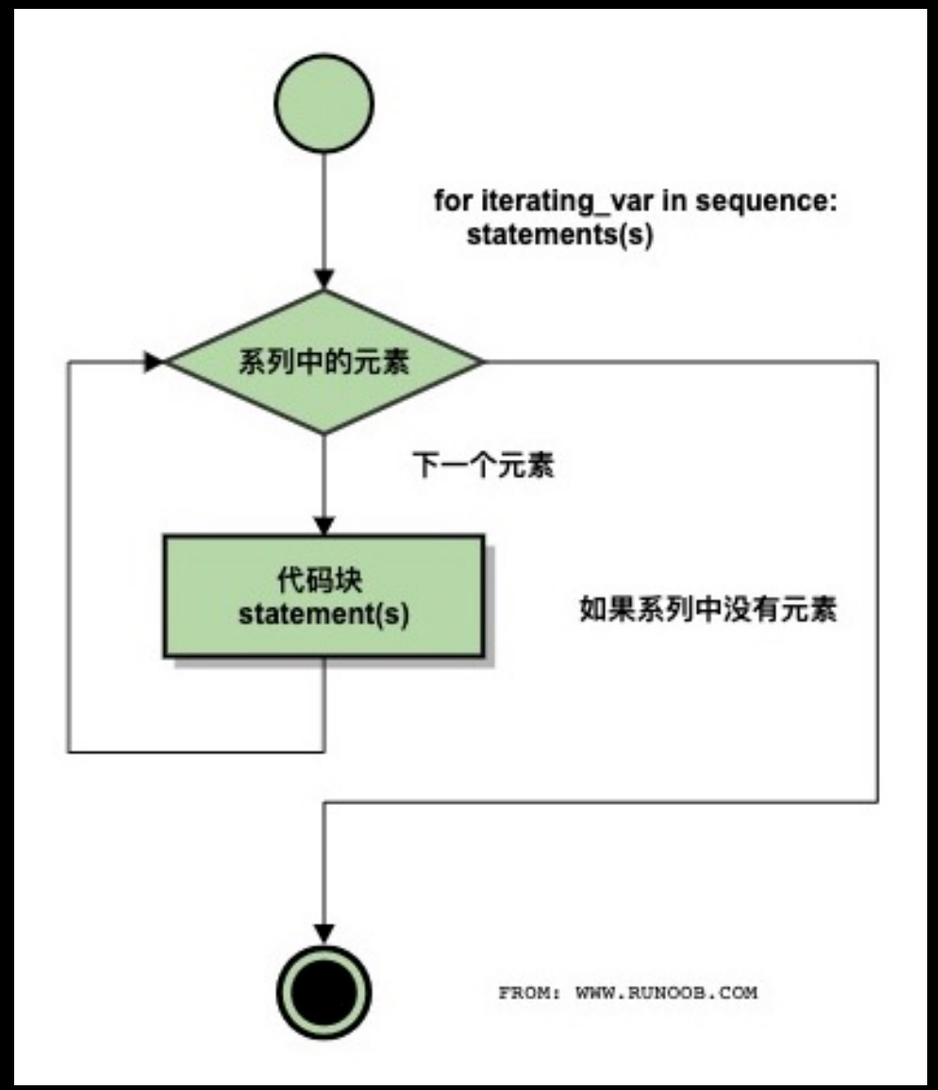
成员包括字符串与各种容器

运算符	描述	实例
in	如果在指定的序列中找到值返回 True，否则返回 False	x 在 y 序列中，x in y 返回 True
not in	如果在指定的序列中没有找到值返回 True，否则返回 False	x 不在 y 序列中，x not in y 返回 True

for循环

for循环可以遍历任何序列的项目，如一个列表或者一个字符串。

```
for iterating_var in sequence:
    执行语句.....
```



break语句

break语句用来终止循环语句，即循环条件没有False条件或者序列还没被完全递归完，也会停止执行循环语句。
break语句用在while和for循环中。
如果您使用嵌套循环，break语句将停止执行最深层的循环，并开始执行下一行代码。

```
for letter in 'qbcnhyrti':
    if letter == 'h':
        break
    print(letter)

var = 10
while var > 0:
   print(var)
   var = var -1
   if var == 5:   #当变量var等于5时退出循环
      break
```

continue语句

continue语句用来告诉Python跳过当前循环的剩余语句，然后继续进行下一轮循环。
continue语句用在while和for循环中。

```
str = ""
for letter in 'fjasdhfhfhjhddh':      # 打印所有不是h的字母
    if letter == 'h':
        continue
    str = str + letter
print(str)

var = 10                              # 第二个实例
while var > 0:
    var = var -1
    if var == 5:
        continue
    print('当前变量值 :' + var)
print("Good bye!")
```

range用法

range()函数可创建一个整数列表，一般用在for循环中。

函数语法：

```
range(start, stop, step)
```

参数说明：

start:计数从start开始。默认是从0开始。例如range(5)等价于range(0, 5)

stop:计数到stop结束，但不包括stop。例如：range(0, 5)是[0, 1, 2, 3, 4]没有5

step：步长，默认为1。例如：range(0, 5) 等价于range(0, 5, 1)

函数

函数是组织好的，可重复使用的，用来实现单一、或相关联功能的代码段。

语法：

```
def functionname(parameters):
    function_suite
    return [expression]
```

```
# 定义函数
def print_me(str):
   print(str)

# 调用函数
print_me(“我要调用用户自定义函数!")
print_me(“再次调用同一函数")
```

return语句

return语句[表达式]退出函数，选择性地向调用方返回一个表达式。

```
total = 0 #这是一个全局变量
def sum( arg1, arg2 ):
    #返回2个参数的和
    total = arg1 + arg2 #total在这里是局部变量.
    print("函数内是局部变量 : " + total)
    return total

#调用sum函数
sum( 10, 20 )
print("函数外是全局变量 : " + total)
```

字符串的格式化

str.format()

我们以前用+连接字符串和数字

例如：banana_count = 3

“we have ” + str(banana_count) + “bananas.”

用format函数我们可以这么做

```
apple_count = 5
“we have {} bananas and {} apples. ”
.format(banana_count, apple_count)
```

我们还可以在{}中放入数字来指定参数位置

"{1}{0}{1}".format("hello", “world")的结果为

'world hello world’

还能用关键字指定

```
“My name is {name}, i’m {age} years old.”
.format(name = “kinkin”, age = 2)
```

字符串的一些操作

str = ‘Hello!’

- len(str)

可获取str的长度 str的长度为6

- str.replace(str1, str2, num)

把字符串中的str1替换成str2,如果num指定，则替换不超过 num次.

str.replace(‘H’, ‘a’) 把str中的H替换为a，字符串变成了aello!

- str.split(str1)

以str1为分隔符切片str

str = ‘hello.txt’

strs = str.split(‘.’)

str[0]的值为hello，而str[1]的值为txt

str.lower()可以将str中的字符变为小写

str.upper()则相反

练习

- 打印200以内所有偶数
- 实现replace
- 实现lower与upper
- 递归解决斐波那契
- 将所输入的字符，以相反顺序打印出来