### 第一节：

<http://www.cnblogs.com/wupeiqi/articles/4906230.html>

file = file(‘test.txt’,’r+’)

for I in xreadlines（）:

print i

for I in file:

print i

### 第二节：

http://www.cnblogs.com/wupeiqi/articles/4911365.html

##### 作用域：

只要内存中存在 则就能被使用

##### 三元运算：

格式：Result1 if 条件 else Result2

举例：

如果条件为真则返回if前的结果如果不为真则返回else后的结果

#-\*- coding: utf-8 -\*-  
name = 'Alan'  
print 'Alan' if name == 'Alan' else 'other'

输出：

Alan

#-\*- coding: utf-8 -\*-  
name = 'Alas'  
print 'Alan' if name == 'Alan' else 'other'

输出：

Other

对象：

Python 的一切事物都是对象 对象是基于类创建的

比如 123456 属于int类

比如 string 属于 str 类

比如 12.5 属于浮点 类

….

type（types）查看对象类型

dir（类名）查看类的所有方法

help（查看类的详细信息 包括所有方法的详解）

help（类名.方法） 查看类下的一个方法的详细介绍

##### 内置类型：

###### 数字类型：

**int （type） 将类型转换为整数型**

>>> int('12')

12

**int(‘value’,base=type) 将值按照base（base 可以为 8 16 2 10）指定的进制转换为十进制**

>>> int('F',base=16)

15

abs(value) 取绝对值

>>> abs(-6)

6

>>> abs(6)

6

**cmp(value1,value2) 比较大小 如果value大于value则返回1小于返回-1 等于返回0**

>>> cmp(12,11)

1

>>> cmp(12,13)

-1

>>> cmp(12,12)

0

**divmod（value，value1） value除以value1 将商和余数作为一个元祖输出**

>>> divmod(15,2)

(7, 1)

**float（type） 将类型转换为浮点型**

>>> float(13)

13.0

###### 字符类型：

**string.capitalize() 字符串首字母大写**

>>> string = 'hello'

>>> string.capitalize()

'Hello'

**string.center（20,’-’） 总长度为20个字符居中显示string两边填充- （如果不指定填充符则**默认 填充空格）

>>> string = 'hello'

>>> string.center(20)

' hello '

>>> string.center(20,'-')

'-------hello--------'

**string.count（s，start，end） 统计s在string中从下标start开始的个数找到end位为止 （下标位可以自己指定 如不指定则默认从0开始 到最后一位结束）**

>>> string = 'welcome python gogogo'

>>> string.count('o')

5

>>> string.count('o',6)

4

>>> string.count('o',4,6)

1

**string.endswith(‘s’) 判断字符串是否以s结尾 如果是则返回True 否则返回False**

>>> string = 'hello word'

>>> string.endswith('h')

False

>>> string.endswith('d')

True

**string.expandtabs（tabsize） 将table转换为空格 空格个数为tabsize，tabsize不指定默认8个空格**

>>> string = 'hello word'

>>> string.expandtabs()

'hello word'

>>> string.expandtabs(1)

'hello word'

>>> string.expandtabs(0)

'helloword'

**string.find(‘s’,start,end) 查找字符串s的下标位存在返回下标 不存在返回-1 start、end 查找的开始位和结束位（不指定start默认起始0 不指定end则结束位为最后一位）**

>>> string = 'let gogogo'

>>> string.find('g')

4

>>> string.find('g',7)

8

>>> string.find('g',5)

6

>>> string.find('g',7,9)

8

***（注：find只打印第一个匹配值的下标 如果不指定start则每次都是匹配第一个匹配的值）***

**string.format（）格式化字符串将值动态传入字符串**

format通过下标、变量两种方式传值

下标：

>>> string = 'My name is {0} , age {1}'

>>> string.format('Alan',25)

'My name is Alan , age 25'

变量：

>>> string = 'My name is {n} , age {a}'

>>> string.format(n='Alan',a=25)

'My name is Alan , age 25'

***（注：下标是有序的必须根据下标位来对应相应的值 变量无序根据变量名匹配）***

***string.join()***

***string.ljust (longs，‘value’) 左对齐***

***string.lower() 字符串变小写***

***string.upper() 字符串变大写***

***string.swapcase() 大写变小写小写变大写***

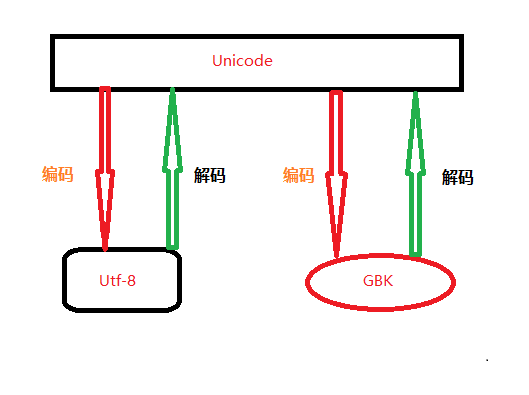
***string.replace（source，des）***

***string.split（）***

***string.index（）***

***string.isalnum（） 判断是否为字母或者数字 不是返回Flash 是返回True***

解码编码：



如上图所示 UTF-8和GBK 不能直接转换 必须从一种编码格式（比如 UTF-8）解码为Unicode然后再从Unicode编码成另一种编码格式（比如 GBK）

编码和解码分别通过encode 和 decode 两种方法实现

编码：string.encode

解码：string.decode

string.decode(‘原编码’) 将原编码类型的字符串解码为unicode

>>> string = '无'

>>> string.decode('utf-8')

u'\u65e0'

注：上面起始位的u为unicode编码的标示

string.encode(‘要编码成的编码’)

>>> string = '无'

//先将utf-8的字符串解码 然后编码为gbk

>>> string.decode('utf-8').encode('gbk')

'\xce\xde'

##### 内置方法：

函数：

方法1.内置方法执行方式可能有多种执行方式 对象.方法

方法2.非内置方法执行方式只有一种

**列表：**

list.count(‘s’) 元素出现的次数

list.pop([index]) 删除并返回下标的值 如果没有指定下标则删除最后一个

list.remove（value） 删除值 如果多个相同值则删除第一个

list.reverse（） 翻转列表

**字典：**

dic.clear（）清除所有内容

dic.copy（） 浅copy

dic.get(‘key’,’argv’) 获取字典value argv当get不到值的时候返回该值

dic.fromkeys(list,values) 列表中的每个值取出来做key每个key的value为values

dic.has\_key（key） 判断字典中是否存在这个key

dic.pop（key） 删除字典中的匹配key的元素

c.setdefault(key,value) 如果值存在则返回如果不存在则创建

dic.update(d) 两个字典整合成一个 如果d字典中和a中的key冲突则d的key覆盖a

**集合：**

set() 过滤掉重复 设置成为集合

>>> subset=set([1,1,2,3,4,4,6])

>>> subset

set([1, 2, 3, 4, 6]) //集合中剔除了重复的值 这里剔除了1,4

>>> type(subset) //查看subset类型为set集合

<type 'set'>

subset\_1 &subset\_2 求交集

>>> subset\_1 =set([1,2,3,4,5,6])

>>> subset\_2 =set([1,3,4,6])

>>> subset\_1 & subset\_2

set([1, 3, 4, 6])

subset\_1 | subset\_2 求并集

>>> subset\_1 =set([1,2,3,4,5,6])

>>> subset\_2 =set([1,3,6,9,10])

>>> subset\_1 | subset\_2

set([1, 2, 3, 4, 5, 6, 9, 10])

subset\_1 ^ subset\_2求反交集

>>> subset\_1 =set([1,2,3,4,5,6])

>>> subset\_2 =set([1,3,6,9,10])

>>> subset\_1 ^ subset\_2

set([2, 4, 5, 9, 10])

subset\_1 – subset\_2 剔除所有set2集合中有的值

>>> subset\_1 =set([1,2,3,4,5,6])

>>> subset\_2 =set([1,3,6,9,10])

>>> subset\_1 - subset\_2

set([2, 4, 5])

subset\_1.issubset(subset\_2) 判断subset\_1是否为subset\_2的子集

>>> subset\_1 = set([1, 2, 3, 4, 5, 6])

>>> subset\_2 = set([1, 3, 4])

>>> subset\_1.issubset(subset\_2)

False

>>> subset\_2.issubset(subset\_1)

True

subset\_1.issuperset(subset\_2) 判断subset\_1是否为subset\_2的父集

>>> subset\_1 = set([1, 2, 3, 4, 5, 6])

>>> subset\_2 = set([1, 3, 4])

>>> subset\_1.issuperset(subset\_2)

True

set.remove(value) 删除集合set中的value值

>>> subset\_1 = set([1, 2, 3, 4, 5, 6])

>>> subset\_1.remove(4)

>>> subset\_1

set([1, 2, 3, 5, 6])

subset.update（value）更新集合将value更新到集合

>>> subset = set([1,2,5])

>>> subset.update([5,6,7])

>>> subset

set([1, 2, 5, 6, 7])

### 第三节：

<http://www.cnblogs.com/wupeiqi/articles/4943406.html>

#### 1.collections 计数器

collection.counter(‘string’)

collection.counter(‘string’)

.update

#### 2.有序字典

#### 3.默认字典：

collection.defaultdict(list) 指定dict的value 默认值的类型

#### 4.可命名元组

>>> Mytuple = collections.namedtuple('Mytuple',['x','y'])

>>> new = Mytuple(1,2)

**>> new**

**Mytuple(x=1, y=2)**

#### 5.队列：

que=collection.deque（）

##### 一、双向队列

特性：

两边可取可插

线程安全 一个线程拿到加锁别的动不了

##### 二、单向队列

import Queue

Queue.Queue.

队列：先进先出

栈：后进先出

#### 冒泡算法：

#### 内置函数：

##### vars（） 获取当前模块中有所有变量

##### \_\_file\_\_ 获取当前模块的路径（相对路径）

##### \_\_doc\_\_ 获取当前模块的注释 （文件头部“”“ 引号注释部分）

##### \_\_name\_\_ 如果被执行的脚本 \_\_name\_\_ 的值为\_\_main\_\_

##### reload（）重新加载模块（更新模块）

##### divmod（）

##### all() 接受一个序列判断如果所有值都是真则返回真 否则则返回假 用途：判断是否有空值

##### enumerate（list,1） 返回下标和值 后面参数定义下标开始位

>>> for k,v in enumerate(list\_tmp):

... print k,v

...

0 1

1 2

2 3

##### chr() 输入数字返回asici码中对应的字符串

##### oder() 输入字符串返回asici码中对应的数字

#### 函数：

##### 函数的参数

###### 普通参数：

形参：

定义一个函数 声明一个参数 这个参数是形式上的 称为形参

实参：

实际执行函数的时候传递的参数称为实参

###### 默认参数：

函数中给参数指定一个默认值 如果调用函数的时候指定了参数则使用指定参数 如果未指定则使用默认参数

默认参数必须放在参数的最后 默认参数可以有多个

###### 动态参数：

\*argv 传入一个元组

\*\*argv 传入一个字典

##### 函数的返回值 return 返回值并退出函数 如果未指定返回值则返回None

def function():

return value

### 第四节：

http://www.cnblogs.com/wupeiqi/articles/4963027.html

### 第六节：

1. json/pickle

作用：

为什么要用json？

硬盘之接受字符串和二进制， 如果将复杂的数据结构（比如：字典、元组、列表）写入硬盘就必须要将其转换为字符串然后才能存到硬盘，由于不同程序 不同语言之间有的时候要进行内存交互 然而不同程序的内存是相互独立的 要进行数据交互就必须要用一个中间的插件做转换 这个插件就是json。

作用一：

不同不同语言、不同平台之间进行复杂格式交互

作用二：

多程序之间内存数据交换

主要操作文件

json.dump() 序列化

json.load()

主要处理临时数据

json.dumps()

json.loads(string)