# Python

## Pycharm的快捷键

Alt + Enter：快速修正

Ctrl + Alt + L：代码格式化

Ctrl + y：删除选定的行

## If else，elif

If a>b:

elif a=b:

else :

## range

range是python的内置函数，range有三种方式构造

1，（stop） 2，（start,stop）3，（start,stop,step）

R=range(10)是指数组从0到9。

R=range(1,10)是指数组从1到9

R=range(1,10,2)是指数组p[1,3,5,7,9]

要想打印出R 要用到list列表。如print(list(R))

not in和in是判读某个数是否在range里面

r=range(1,10,2)  
print(list(r))  
print(10 not in r)  
print(9 not in r)  
print(10 in r)  
print(9 in r)

## while

while 判断：

执行语句

## for in

## for变量名 in 要依次遍历的东西

注：in后是要可迭代的，如字符串和列表

·······························for item in 'qdwe':  
 print(item)  
for item2 in str(21311):  
 print(item2)  
for \_ in range(5):  
 print('python')  
sum = 0  
for item3 in range(1, 101):  
 sum += item3  
print(sum)

## 

## else

else不仅可以和if连用，还可以和while，for连用

a = 0  
while a < 3:  
 pwd = input('输入，密码:')  
 if pwd == '8888':  
 print('密码正确')  
 break  
 else:  
 print('密码不正确')  
 a += 1  
else:  
 print('三次已过')

只要没有触发break，那么在执行完while后就会执行break

for循环也是如此

## index--list列表的内置对象

lst = ['hello', 'world', 98, 'hello']  
print(lst.index('hello'))  
print(lst.index('hello', 1, 4))

返回值是0和3

当list中有多个相同元素时，index默认给出第一个

List中也可以用负号象征的位置如-1，-2等

## list切片

List列表对象的后面直接加上[a：b：c]是指拷列表中相应的值

如[1：3：1]就是将列表中从一号位置的值到3号位之前的值（不包括3），以1步长进行拷贝。

如果没有直接确定步长，默认为1。

没确定初始位，默认为0。

没确定末位置，默认为最后一位。

步长可以为负，这样就是从后往前拷贝

表格

描述已自动生成

其中，切片是替换掉元素

表格

描述已自动生成

## List的内置方法-排序

lst = [12, 37, 25, 98, 7]  
lst.sort() # 升序  
print(lst)  
lst.sort(reverse=True)#降序  
print(lst)  
print('---------------------------------')  
lst = [12, 37, 25, 98, 7]  
lst2 = sorted(lst) # 产生新的列表  
print(lst)  
print(lst2)

列表的生成式

lst = [i for i in range(1, 10)]  
print(lst)  
lst = [i \* 2 for i in range(1, 6)]  
print(lst)

第一个是1到9

第二个是2，4，6，8，10、

## 字典

scores = {'张三': 100, '李四': 98, '王五': 45}  
print(scores)  
print(type(scores))  
'''使用dict（）'''  
student = dict(name='jack', age=20)  
print(student)  
'''空字典'''  
d = {}  
print(d)

代码输出：

{'张三': 100, '李四': 98, '王五': 45}

<class 'dict'>

{'name': 'jack', 'age': 20}

{}

## 字典的查找

scores = {'张三': 100, '李四': 98, '王五': 45}  
'''第一种方式，使用[]'''  
print(scores['张三'])  
'''第二种方式，使用get（）方法'''  
print(scores.get('张三'))  
print(scores.get('陈六'))  
print(scores.get('麻七',99))

代码运行结果

100

100

None

99

如果是print(scores.get('陈六'))的话

那么就会报错，这就是加不加get的区别

scores['wqjh'] = 90

这个代码就是给字典增加内容

del scores[‘张三’]就是将键张三和它的值全部删除

获取key和value

scores = {'张三': 100, '李四': 98, '王五': 45}  
keys = scores.keys()  
print(keys)  
print(type(keys))  
print(list(keys)) # 将所以的key组成的视图转换成列表  
  
# 获取所有的value  
values = scores.values()  
print(values)  
print(type(values))  
print(list(values))  
  
# 获取所有的key-value对  
items = scores.items()  
print(items)  
print(list(items)) # 元组

## 代码运行

dict\_keys(['张三', '李四', '王五'])

<class 'dict\_keys'>

['张三', '李四', '王五']

dict\_values([100, 98, 45])

<class 'dict\_values'>

[100, 98, 45]

dict\_items([('张三', 100), ('李四', 98), ('王五', 45)])

[('张三', 100), ('李四', 98), ('王五', 45)]

## 字典生成式

it = ['a', 'b', 'c']  
pr = [96, 78, 85]  
d = {item.upper(): price for item, price in zip(it, pr)} # upper是为了将所有英文字母大写  
print(d)

## 代码运行

{'A': 96, 'B': 78, 'C': 85}

## 可变与不可变序列

可变：列表，字典 , 集合

不可变：元组，字符串

## 元组的创建

*'''第一：直接加（）'''*'''注：第一种可以不加（）'''  
t2 = 'python', 'java', 'c'  
'''第二：使用内置函数tuple（）'''  
t1 = tuple(('python', 18, 'yes'))  
print(t2)  
print(t1)  
'''当元组只有应该元素是，使用小括号和逗号'''  
t = (10,)  
print(t)

## 集合的创建

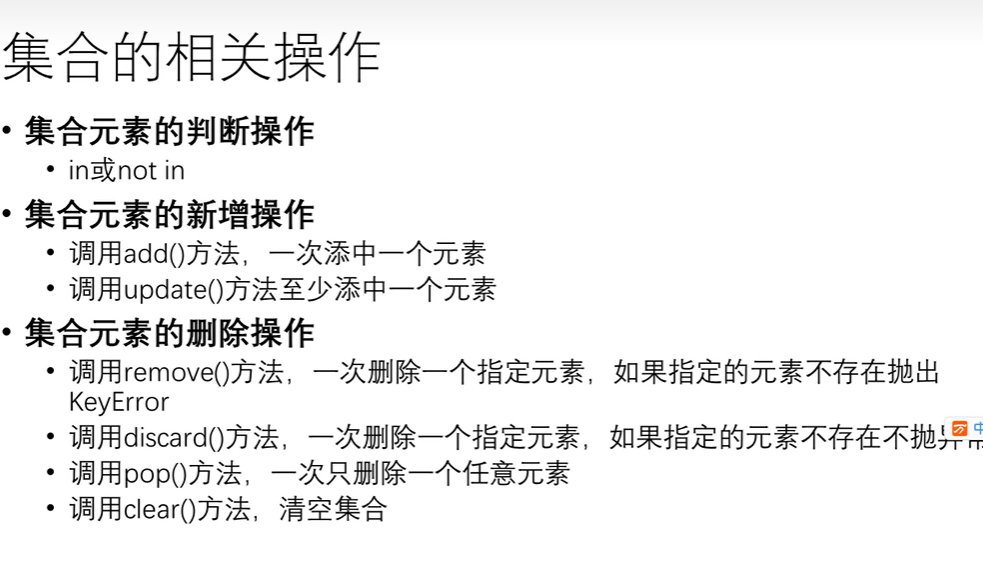
*'''use{}'''*s = {2, 3, 4, 5, 6, 7, 7} # 集合的元素不能重复  
print(s)  
'''use ste()'''  
s1 = set(range(6))  
print(s1, type(s1))  
s2 = set('python')  
print(s2)  
'''空集合的创建'''  
# 必须是set（）创建，不能是s={}这样创建  
s3 = set()

## 运行

{2, 3, 4, 5, 6, 7}

{0, 1, 2, 3, 4, 5} <class 'set'>

{'y', 'o', 'p', 't', 'h', 'n'}



注：集合是无序的，使用add和update的时候，所加的元素不会按照顺序加在集合的最后面,而是使用hash去进行计算

根据上面，判断两个集合是否相等，只要看集合里面的值是否相等，和顺序无关

Pop的是不能加参的，就是pop（）括号里面不能写参数

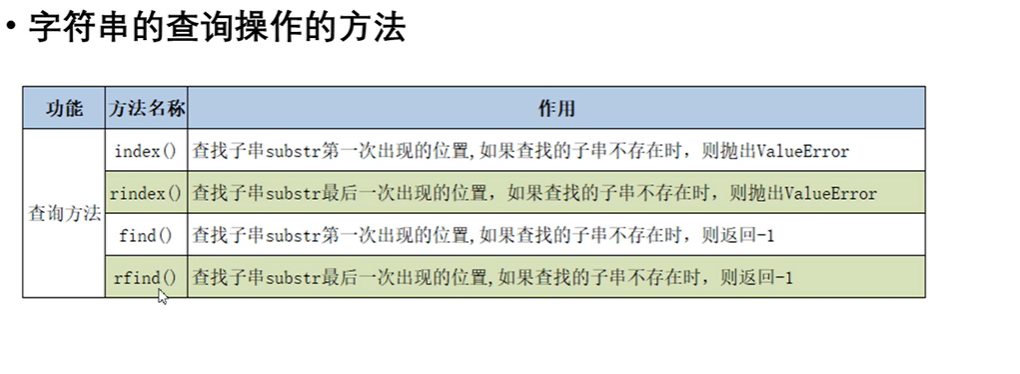


## # 集合生成式

s = {i \* i for i in range(10)}

print(s)表格

描述已自动生成



字符串的劈分

s = 'hello world python'  
lst = s.split() # 默认是以空格为劈分符  
print(lst)  
s1 = 'hello/world/python'  
print(s1.split(sep='/')) # 设置劈分符  
print(s1.split(sep='/', maxsplit=1)) # 设置最大劈分次数

## 运行

['hello', 'world', 'python']

['hello', 'world', 'python']

['hello', 'world/python']

注：split是从左侧开始劈分，rsplit是从右边开始

劈分返回值是列表

## 函数

def fun(a, b, c):  
 print('a=', a)  
 print('b=', b)  
 print('c=', c)  
  
  
fun(10, 20, 30)  
lst = [11, 22, 33]   
fun(\*lst)  
print('----------------------')  
fun(a=100, c=200, b=300)  
dic = {'a': 100, 'b': 200, 'c': 300}  
fun(\*\*dic)

加 \*可以将列表里面的数传进来，加\*\*可以将字典里面的数传进来

## 异常处理机制

## 

try:  
 a = int(input('给一个整数'))  
 b = int(input('再来一个'))  
 res = a / b  
except ZeroDivisionError:  
 print('wrong')  
else:  
 print('结果为：', res)  
finally:  
 print('finall是无论如何都会执行的')

try里面写的是肯会出错的代码

except经行捕获

else是当try里面的代码没问题是执行的

finally通常是执行代码内存释放这类的代码

## 类的定义

class Student:  
 pace = '吉林'  
  
 def \_\_init\_\_(self, name, age): # 初始化的方法名字最好用\_\_init\_\_(self)  
 self.name = name  
 self.age = age  
  
 def eat(self):  
 print('ass')  
  
 @classmethod  
 def cm(cls): # 要加cls  
 print('类方法')  
  
 @staticmethod  
 def method(): # 不能加self和cls  
 print('静态方法')

## python类的动态绑定属性

接着上面的代码

asd = Student('德玛西亚', 10)  
asd.genger = '女'  
print(asd.genger)

可以直接再实例化之后创建新的属性

方法也是同理，可以直接动态创建

## 如何将类内的属性不给别人访问

class Student:  
 def \_\_init\_\_(self, name, age):  
 self.name = name  
 self.\_\_age = age # 双下划线是在外部无法使用的属性  
  
 def show(self):  
 print(self.name, self.\_\_age)  
  
  
stu = Student('张三', 20)  
stu.show() # 这属于类内  
# 在类外使用age  
print(stu.name)  
# print(stu.age)#报错  
# 可以强行访问  
print(stu.\_Student\_\_age)

## 继承

class Person(object):  
 def \_\_init\_\_(self, name, age):  
 self.name = name  
 self.age = age  
  
 def info(self):  
 print(self.name, self.age)  
  
  
class Student(Person):  
 def \_\_init\_\_(self, name, age, score):  
 super().\_\_init\_\_(name, age)  
 self.score = score  
  
  
stu = Student('jack', 10, 90)  
stu.info()

当然也可以经行方法的重写

## 多态

Python作为一个动态的语言，可以不那么明确自己的继承关系

图示

描述已自动生成

上面代码中的person类是继续于object的，但是还是能够使用eat

