# Python学习笔记

## 语言点

java 中的基本数据类型

byte char int short float double boolean String

a = 10

b = 20

a,b = b, a # 这就是交换a，b的值

import math # 注意导包

a = 4

print(math.sqrt(a)) // 2.0

## 数据类型

不可变数据类型 ： Number（int,float,bool,complex,）,String

（python中没有字符）,Tuple

可变数据类型 ： List，Dictionary，Set

Number(int ,bool,float,complex)python中没有double

String ，可以切片，有索引

Truple ，可以切片，有索引，（）为创建的方式，一般函数返回多个值，是

以元组的形式返回的，函数中的可变参数，python会封装为元组

d=1,2,3  
print(d) // (1, 2, 3)

Set ，使用{ }来创建，主要用来去重的操作

List ，可以切片，有索引， [ ]为创建的方式

Directionary，没有索引，使用{ } 来创建，使用[ key ]来获取数据

### 字符串：

s = "abcdefg"

b = ‘a’ 这也是一个字符串，python中没有char类型

print(s) # abcdefg

print(s[0]) # a

print(s[-1]) # g

s[1] = "D" # 这样的操作是报错的

切片：

s = "abcdefg"

print(s[1:3]) # bc 切片的时候，前开后闭

print(s[::2]) # aceg 第三个参数，是步长

print(s[::-1]) # gfedcba ，如果最后一个参数为-1，表示逆序

转义

s = r"abc\ndefg" # r对字符串中的内容进行转义

print(s) # abc\ndefg

不同的类型是不可以相加的

### 元组：

(和字符串有些类似，创建的时候，元组使用的是()创建，列表使用的是[]来

创建)

t = ('a',10,0.3)

print(type(t))

print(t) # ('a', 10, 0.3)

print(t[0]) # a

# t[1] = "A" # 由于truple是不可变的数据类型，因此无法个更改，只能读取，不能删除

print(t[:2]) # ('a', 10)

### 列表

list = [1,2,0.3,"isea",'you']

print(type(list)) # <class 'list'>

print(list) # [1, 2, 0.3, 'isea', 'you']

print(list[1]) # 2 获取列表中的内容

print(list[1:]) # [2, 0.3, 'isea', 'you']

list[0] = "A" # list的修改操作

print(list) # ['A', 2, 0.3, 'isea', 'you']

#

del(list[0],list[1],list[2]) # [2, 'isea']

print(list)

print(len(list)) # 2

print('isea' in list) # True

list = [x for x in range(1,101)] // range(x,y)前开后闭

print(list)

### 字典：

字典：类似java中的map，

d = {'name':'isea','age':30,'sex':'男'}

print(d) # {'name': 'isea', 'age': 30, 'sex': '男'}

print(type(d))

print(d['name']) # isea

d['name'] = '张飞'

print(d['name']) # 张飞

del(d['sex'])

print(d) # {'name': '张飞', 'age': 30}

dic = { # x 是key，x + 1 是 value，key的值是遍历0~5

    x:x+1 for x in range(5)

}

print(dic) # {0: 1, 1: 2, 2: 3, 3: 4, 4: 5}

print(len(dic)) # 5

如何遍历一个字典：

def city\_def(param):  
 for key,value in param.items():  
 print(key,':',value)   
a = {'bj':'32c','sh':'31c','xm':'23c'}  
city\_def(a)  
  
# bj : 32c  
# sh : 31c  
# xm : 23c

print(points[0,0]) 第1行，第一列

print(points[0,1]) 第1行，第二列

x = array(points[:,0]) 所有行，第一列

y = array(points[:,1]) 所有行，第二列

## 逻辑

### 条件

a = 17

if a == 10: # 这里的if后面必须要有一个空格，还要有一个：

    print("hello world...")

else: # 这里也必须要有一个冒号

    print("wxwxwx...")

python中如果是空的话，那么该值就认为是false

if None :

    print("right...") # 无打印

if 0.0:

    print("right...") # 无打印

if []:

    print("right...") # 无打印

if 0:

    print("right...") # 无打印

if "":

    print("right...") # 无打印

if {}:

    print("right...") # 无打印

### 循环：

sum = 0

for i in range(1,101):

    sum += i

print(sum) # 5050

i = 1

sum = 0

while i < 101:

    sum += i

    i += 1

print("sum = ", sum) # sum = 5050

id()的用法

a = 10

b = 10

c = 10.0

print(id(a)) # 1490447664 获取 a 的引用地址

print(id(b)) # 1490447664

print(id(c)) # 1606315876472

print(a is b) # 等价于 id(a) == id(b)

### 函数

解释型语言，先定义后执行

def getName():

    return 'isea',100

print(getName()) # ('isea', 100) 返回值是一个元组

def test(a,b,\*args):

    print(args)

    print(a,b)

test(100,200,1,2,3)

"""

(1, 2, 3)

100 200

"""

## 面向对象

Python 支持多继承

实例变量都是隐式的，显示的变量是类变量

class Animal(object): # object 表示继承object类

    def eat(self):

        print("Animal eating....")

class Person(Animal): # python中支持多继承

    #python中的类变量

    ancestor = "monkey"

    #python中的类方法，添加装饰器注解，和类的生命周期相同

    @classmethod

    def cry(cls):

        print(cls.ancestor,"crying....")

    #python中的静态方法

    @staticmethod

    def test():

        print(Person.ancestor,"this is a static method...")

    #python中的实例变量都是隐式的

    def \_\_init\_\_(self): # self参数必须要给与

        print("一个person被创建了...")

    def \_\_init\_\_(self,username):

        self.name = username

        self.age = 10

    def getInfo(self):

        return self.name,self.age

p = Person("isea") # ('isea', 10) 这里的返回值是元组

print(p.getInfo())

print(Person.ancestor) # monkey

Person.cry() # monkey crying....

Person.test() # monkey this is a static method...

p.eat() # Animal eating....