python

python简介：是一种解释型的 面向对象的高级程序设计语言

解释型：需要有解释器解释的语言

面向对象 对象：具有某些功能的个体

python能做什么

1. 人工智能
2. 自动化测试
3. 爬虫
4. 数据分析
5. 网站后台程序

…………

优点：简单、开源、免费、完善的代码库

缺点：代码不能加密、这行速度慢

python安装（3.7.2版本）

首页 add python to puth 要勾选（自动把安装路径添加到环境变量里）

cmd命令行输入python

报错：python不是内部或外部命令

解决：将python的安装路径手动添加到环境变量里

此电脑 <右键<属性<高级环境设置<环境变量<系统变量<path

python版本

2.x

3.x

两次比较大的版本

最大区别：语句不一样

#!/bin/python

#-\*- coding:utf-8 -\*-

1. 输出（打印）和输出（从键盘上手动输入）

print 输出 py3 print必须加引号

print（“abc”） 字母、字符串加引号

input 输入 表示需要从键盘输入

格式： 变量名=input（“提示语”）

a=input（“请输入用户名:”）

print(a)

2.变量 会变化的量

格式： 变量名=值

变量名的规则：（1）字符、数字、下划线组成

（2）不能以数字开头

（3）不能跟python中的函数相同

打印变量的值，不需要加$

print(abc)

变量的值如果是字符的话，必须加引号

多变量赋值

变量名1,变量名2………= 值1,值2…………

a,b,c=1,2,3

a,b,c=1,2,None :代表c是空值

python解释器（默认cpython）

cpython：将python代码解释成c语言的字节码（c语言开发的解释器）

Jpython：Java语言开发的解释器，将python代码解释成Java的字节码

python：python语言开发的解释器，提高了python代码的执行速度

python语言又被称作胶水语言

数据类型

字符串、列表、元组、字典、集合、整数、浮点数、空值、布尔值

1整数（int）

定义整数 1.变量名=值（整数） 2.变量名=int(值)

abc=123

第一个python的内置函数（type）:查看变量的数据类型

print（type(abc)）

<class’int’>

将其他的数据类型更改为整数（仅限于阿拉伯数字）

bc=12.3

abc=int(bc)

print(type(abc))

<cless’int’>

2浮点数（float）

定义浮点数 1.变量名=值（小数）2.变量名=float(值)

将其他的数据类型更改为小数

abc=12

abc=float(abc)

print(type(abc))

<cless’float’>

3 空值 特殊的值

None

4 布尔值（bool）

True为真 False为假

5 字符串(str) 一串字符的集合

定义字符串 1.变量名=‘值’（任意）2.变量名=str（ ’值’）

特点：1.不可变的 2.支持索引 3支持切片

变量名=值（不可变）

abc=’a’+’1’

print(abc)

结果：a1

根据下标位置取对应的元素叫索引

首位从0开始

索引格式：变量名[下标位置]

abc=‘qwer’

print(abc[1])

结果：w

支持反索引

abc=‘qwer’

print(abc[-1])

结果：r

切片：变量名[下标值1:下标值2]

取值范围：从1到2结束 不取2位置上的值

abc=‘qwer’

print(abc[1:3])

结果：we

变量名[下标值1：]

取值范围：冒号在前代表最前 冒号在后代表最后

abc=“qwer”

print(abc[:2])

结果：qw

abc=“qwer”

print(abc[1:])

结果：wer

取值范围：：冒号在后表示最后

在前表示以这个数字的大小分组 取组中的第一个

abc=“qwer”

print(abc[::2])

结果：qe

print(abc[::3])

结果：ar

递进:abc=“1234567”

print(abc[0:5:2)

结果：135

属于字符串的函数

1. 将字符串小写变大写 upper() 格式：变量名.函数名()

abc=“qweere”

f=ab.upper()

print(f)

结果：QWEERE

2．将字符串的首字母变大写 capitalize()

3.将字符串大写变小写 小写变大写 swapcase()

4.统计字符串中某个元素有多少个 count(“”)

5.获取某个元素的下标号 index(“”) 相同的话只获取第一个

6.将字符串中的某个元素替换 replace(“”，“”)

replace(“”，“”，1)

数字是几替换几个

7.判断字符串是否以某个元素开头 startswith(“”)

8.判断字符串是否以某个元素结尾 endswith（“”）

9.填充字符串（格式化字符串）

1.用大括号在字符串占位置

abc=“1+{}={}”

f=ab.format(“abc”,“789”)

print(f)

结果：1+abc=789

1. 用%s在字符串中占位置

ab=“1+%s=%s”

f=ab%(2,3)

print(f)

结果：1+2=3

%s能填充任意字符

%d只能填充整数

10.去除字符串的空格

strip() 去除左右的空格

lstrip() 去除左边的空格

rstrip() 去除右边的空格

11.将字符串变成列表

以某个元素为分隔符 将字符串变成列表

split(“”)

第二个python的内置函数(len(变量名)):统计变量有多少个元素

len(ab) 统计变量有多少个元素

ab.count 统计某个元素有多少个

内置函数：作用于整个python

字符串的函数：只作用于字符串

列表(list) 一组数据的集合（等同于shell上面的数组）

格式：以中括号为标识，每个元素之间用逗号隔开

定义列表：1.变量名=[“值”,值] 2.变量名=list(“值”)

注：值是有序列的

特点：1.可变的（可以删除 可以增加） 2.支持索引 3.支持切片

abc=[12,“qwe”,“q@#$”,[123,456,789]

print(abc[3][0][0])

结果：1

列表的函数

1. 给列表中添加新的数据（只能添加一个 默认最后一位）

abc.append(10)

print(abc)

1. 可以添加到任意位置（只能添加一个）

insert(,) 第一个：添加的位置 第二个：添加的数据

1. 删除某个元素（只能删一个）

remove()

1. 删除某个元素（通过下标位置）

pop( )

1. 排序（将列表按升序排列）

注：列表中的所有元素是同一种数据类型

字符：按字符中的第1个元素ASCII中的位置

sort()

1. 反转（将列表从后往前显示）

reverse()

1. 统计列表中某一个元素有多少个

abc=[1,2,3,4,5,6,7]

f=abc.count(1)

print(f)

结果：1

1. 获取列表中某个元素的下标

f=abc.index(1)

print(f)

1. 更新（将另外一个列表中的所有元素添加到列表中）

extend

1. 复制

abc=[1,2,4,8,7,1]

bc=abc.copy()

print(abc)

print(bc)

浅复制：只复制了第一层的数据，共用了深层次的数据

深复制：完全复制

import copy

abc=[1,2,[“qwe”,“rty”],90]

bc=copy.deepcopy(abc)

abc[2].append(100)

print(abc)

print(bc)

结果：[1,2[“qwe”,“rty”，100],90]

[1,2[“qwe”, “rty”],90]

统计列表中有多少个元素 len()

元组 tuple 一组数据的集合（等同于shell上的数组）

格式：以括号为标识 每个元素之间用逗号隔开

定义元组：1.变量名=（“值”，“值”）2.变量名=tuple(“值”)

特点：1.不可变的2.支持索引3.支持切片

1. abc=（12,3,4）
2. abc=tuple(12,3,4)

如果元组中只有一个元素时，要加逗号

1. 统计元组中某一个元素有多少个

abc=(12,3,4)

f=abc.count(12)

print(f)

1. 获取元组中某个元素的下标

f=abc.inden(12)

print(t)

字典 dict

以大括号为标识

元素是键值对的方式（key:value）

格式：变量名={key:value}

特点：可变的（可以添加 可以删除）

无序的

不支持索引

注：字典中的key(键)必须是唯一的

支持通过索引键来得到值

字典的函数

1. 添加（向字典中添加键对值）

abc[“age”]=123

print(abc)

如果有就修改，如果没有就添加

即是修改又是添加

1. 删除

abc.pop(“键名”) pop本身会保存删除的信息

abc.popitem 默认 删除最后一个 不保存

1. 获取所有的键 或者所有的值

print(abc.keys())

print（abc.values()）

获取所有的键值对

print(abc.items())

1. 更新 将bc字典中所有的键值对更新到abc字典中

abc.update(bc)

print(abc)

结果就是将bc字典中的所有键值对更新到abc字典中

集合（set）

以大括号为标识 中间用逗号隔开

定义集合：1.变量名={值，值} 2.变量名=set(值)

注：值只能有一个元素（第2种方式）

注：如果定义一个空的集合，不能使用大括号，只能用a=set()

特点：1.无序的 2.不可重复的（集合有两个，只能打印1个） 3.不支持索引 4.可变的

集合的函数

1. 添加

abc.add(110)

print(abc) 如果之前有，那么只能留一个

1. 删除

abc.pop() 默认删除最后一个

print(abc)

abc.remorve(123) 删除指定元素 想删哪个删哪个

print(abc)

运算符

算术运算符

+ 加 - 减 \* 乘 //整除 %求余

a=3+2 print(3+2)

print(a) 结果5

比较运算符

< >= <= !=(不等于) ==(等于)

a=3 赋值

a==3 判断  
逻辑运算符

与 and 或 or 非 not

赋值运算符

+= 加 -= 减 \*= 乘 /=除 //= 整除 %= 求余

a=3

a+=1 等于a=a+1

print（a）

成员运算符

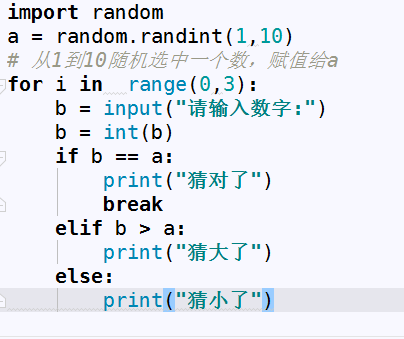
in(在…………之间) not in （不在……………里面）

判断语句

if语句

注： 语句后要加冒号

执行语句块之前必须缩进



单分支 if 条件:

缩进 执行语句块

双分支 if 条件:

缩进 执行语句块

else:

缩进 执行语句块

多分支 if 条件:

缩进 执行语句块

elif 条件:

缩进 执行语句块

else:

缩进 执行语句块

注：input接收到的任何数据都是字符串

嵌套判断：if条件: if条件:

循环语句（for while）

for 格式：for 变量名 in 某一个范围:

执行语句

范围：指的是一组数据的集合（列表、元组、字典 字符串 ）（列表、元组、字典用来存储数据的）

循环里面的第一个函数

range(范围)

range(100)

1. 括号中有1个数字时，表示默认从0开始 到此数字结束 取不到此数字
2. 有两个数字时，表示从第1个数字开始到第2个数字结束，取不到第2个数字
3. 有3个数字时， 第3个数字表示递进

break 中止循环

continue 跳过本次循环

嵌套： 循环中嵌套判断

for i in 范围:

if 条件:

执行语句

elif

else

循环嵌套循环 for语句中加入for语句

大循环执行1次，小循环执行一轮

end=’\t ’不换行

for…………else………语句

原理：只要循环没有被break掉，就执行else语句

enumerate(列表or元组or字符串)

for I in enumerate

作用：将下标位置与数据一一对应

循环语句：while循环

格式：while 条件：执行语句

while 条件:

缩进 执行语句块

for 适用场景：通常用来循环有序列的数据（有序列的是指有下标位置的）

while适用场景：1.无限循环 2.根据条件进行循环时

循环嵌套： while…………while……………

while…………for………………

while…………if…………………

大循环执行一次，小循环执行一轮

while……………else…………语句

原理：只要循环没有被break，就执行else语句

异常捕获语句

异常：因代码的逻辑或者语法上的错误，而导致的程序中断

异常捕获：对异常进行预防

try…………except…………语句

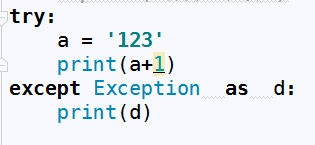
try:

a=’123 ’

print(a+1)

except: 注：默认检测到所有的报错（Exception）

print(‘hello’)



except后跟某一类型的错误：只检测这一类型的错误

原理：将可能会发生异常的语句，放在try语句下面

如果异常发生，程序不会中断，进而执行except下面的语句

except Exception:（可以打印报错的描述）

报错

1. 报错的位置
2. 报错的语句
3. 报错的类型和描述

报错：TypeError:1.不同的数据类型作运算

2.数据类型不能更改为对应的数据类型

NameErroe:（命名错误） 1.变量名没有赋值就使用

SyntaxError:（语法错误）代码的逻辑或者语法不符合规则

try………except…………else………

原理：只要try语句没有错误 才执行else语句

try………except…………finally……………

原理：不论执行try还是except 都执行finally

自定义异常（也叫触发异常） 通常用于开发

raise

例：raise TypeError(‘123’)

python的内置函数 sum(求和) max(求最大) min(求最小) 只能列表、元组

例：a = [1,2]

print(sum(a))

结果：3

对文件的操作（对文件的读、写）

open() 打开

例如：



1. 文件名
2. 如果操作的是当前路径下的文件，直接写文件名
3. 如果是其他路径下的文件，要加上路径 路径不要转义

方法：1.双反斜杠不转义 2.在路径前加r

1. 权限

w 写 write (‘字符串’)

变量名.write(‘abc’)

不具有换行功能 覆盖的模式

r 读 read() 读取文件中的所有内容 结果是字符串

readlines() 读取文件中所有内容，结果是列表，每一行就是列表中每一个元素

readline（）每次只读取文件中的一行，本身具有迭代功能

注：每次执行结束后要关闭文件

变量名.close（）关闭文件

批量操作 for循环

a 写 追加模式的写 write (‘字符串’) 追加的模式

w+ r+ a+ 可读可写可追加

wb rb ab 以二进制的方式读取、写入

函数：可重复使用的具有某种功能的代码块

好处：1.节省代码 2.重复使用

定义函数的格式： def 函数名（）

缩进 代码块

调用函数：函数名（）

函数名的命名规则：

1. 字符串、数字、下划线组成
2. 以字符串、下划线开头
3. 不能以python中的内置函数命令
4. 不能以python中的文件命名

加括号代表执行此函数 不加括号代表函数名

1. 变量的作用域：局部变量和全局变量

在函数里面定义变量是局部变量（在局部范围内有效的）

在文件中定义的变量是全局变量（在全局范围内有效的）

global 变量名 将局部变量变为全局变量

1. 结果赋值

格式:return 值

1. 赋值 将return后面的数据赋值给调用者

2.结束标志 看到return表示函数到此结束

参数传递 （给函数中传入参数）

参数名的命名要符合变量名的规则

1. 必须参数(位置参数) 必须传入的参数

格式： def 函数名（参数名）:

1. 默认参数 可传入也可不传入

格式： def 函数名（参数名=值）:

3.可变长参数 可以传入多个参数

格式： def 函数名（\*参数名）: \*代表多个

原理：接收到多个数据以后，变成元组传到函数中

格式： def 函数名（\*\*参数名）:

原理：接收到的数据格式是键值对

连用：优先级：必须参数>默认参数>可变长参数

python的内置函数：help 查看某些数据类型的函数

4匿名函数 lambda

格式：函数名=lambda：表达式

匿名函数属于自定义函数的一种，lambda和def

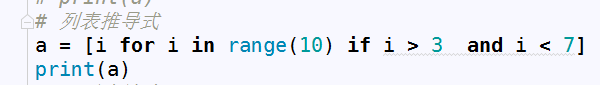
lambda一般用于只有一个表达式的函数，不能处理复杂的、具有逻辑结构的代码 def 能够处理任何的代码

函数名 = lambda 参数名，参数名：表达式

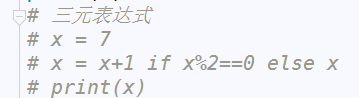
列表推导式

将控制语句写在列表中，使产生的结果直接存在列表中

格式：列表变量名 =[ 变量名 控制语句]



三元表达式



python的内置函数：ord 将字符串转化为ascii中的位置

chr 将数字转化为ascii中的字符

divmod 整除求余

hex(123) 十进制转换十六进制

oct(123) 十进制转换八进制

bin(123) 十进制转换二进制

int(十六进制数、八进制数、二进制数 ) 任何进制转换十进制

join( ) 以字符串为间隔，将括号内连接起来

with 语句 上下文管理器

格式： with 操作的东西 as 变量名：

例：with open（‘a.txt’，‘r’，encoding=‘utf-8’）as f:

以读取的权限打开a.txt这个文件

原理：\_ \_enter\_ \_ 进入 \_ \_ exit\_ \_ 退出

开场前准备 清理工作

保证不管在处理过程中是否发生错误，或者发生异常都会执行 \_ \_ exit\_ \_（清理）操作，释放被访问的资源

访问数据库、socket 、对文件的操作等，需要执行关闭命令的场景

导入语句 import语句

将其他文件的代码导入到本文件中使用

其他文件：模块

random模块的功能：产生随机数

import语句

格式：第一种 import 文件名 （不要加后缀）

导入的是文件中的所有代码 可以使用文件中的任何函数

调用导入模块中的函数：文件名.函数名（）

print(day4.sum(12,10)

第二种 from 文件名 import 函数名

导入的是文件中的某个函数 只能使用导入的那个函数  
调用导入的模块中的函数：函数名（）

from 文件名 import \* \*代表导入这个文件中的所有函数

例：from dag4 import sum as dd

print(dd(12,23))

as:为文件、函数重命名

if \_ \_ name \_ \_ ==‘\_ \_ main\_ \_’:

判断当前执行的文件是否是本文件，如果是 \_ \_name\_ \_ 等于\_ \_ main\_ \_

下载和安装网上模块的方式

1. pip安装 pip是python自带的一个下载和安装模块的组件 命令：在cmd下：pip install 模块名
2. pycharm 工具安装

file – setings – project – project Interpreter – 找到+号下载

1. 文件安装（没网络的情况下）
2. 到网络上下载模块文件

pip install 文件名.whl

python setup.py install

对Excel表格的操作

xlwt(写入) xlrd(读取) xlutils(追加)

写入

import xlwt

#打开一个Excel表格文件（定义编码方式）

f = xlwt.Workbook( encoding=‘utf-8’)

#添加一个标签页（标签页的名字）



sheet=f.add\_sheet(‘excel学习’)

#给sheet标签页写入

#write(行,列,内容) 0,0表示第一行第一列

sheet.write(0,0,‘河南’)

sheet.write(4,0,‘博文’)

#保存

f.sava(‘软件测试.xls’) excel后缀是.xls

读取

import xlrd

打开读取的文件(excel表名)

f = xlrd.open\_workbook(‘软件测试.xls’)

#获取有多少个标签页

#加print是为了自己能看出来

print(f.nsheets)

#获取所有标签页的名字

print(f.sheet\_names())

#1.根据索引来获取想要操作的标签页

sheet = f.sheets()[0] 读取的是标签页

#2.根据标签页的名称获取想要操作的标签页

new\_sheet=f.sheet\_by\_name(‘txt数据’)

获取标签页中有多少行

print(sheet.nrows)

获取第几行的内容

print(sheet.row\_values(1)) 获取第2行的内容 ,从0开始，0表示第1行

#获取标签页中有多少列

print(sheet.ncols)

#获取第几列的内容

print(sheet.col\_values(0)) 0表示第1列,索引从0开始

#获取某个单元格的内容

print(sheet.cell(1,0).value) 表示第2行,第1列 从0开始 0,0表示第1行第1列

追加

import xlrd

from xlatils.copy import copy

#打开一个Excel文件

f = xlrd.open\_workbook(‘软件测试.xls’)

new\_f = copy(f)

#通过索引获取要操作的标签页

sheet=new\_f.get\_sheet(0)

#写入excel文件

sheet.write(0,2,‘河南’)

new\_f.save(‘软件测试.xls’)

os模块 python自带的模块

作用：与操作系统交互

**import** os  
*# os.getcwd() # 获取当前所在的位置  
# 获取某目录下的所有文件  
# os.listdir()  
# 如果加路径，获取此路径下所有文件  
# a = os.listdir(r'D:\Users')  
# print(a)  
# 更改文件名(目录也可以更改)  
# os.rename('day1.py','day10.py')  
#切换目录  
# os.chdir(r'/Users/zhaoshuai/Desktop')  
#创建目录  
#os.mkdir('目录名')  
#创建递归目录  
# os.makedirs('bbb\ccc')  
#删除空目录  
# os.rmdir('目录名')  
#删除递归目录  
# os.removedirs('bbb\ccc')  
#popen 执行命令(直接有结果的命令)  
# a = os.popen('ping 192.168.0.1')  
# print(a.read())  
#将路径与文件分隔开(分隔的只是字符串,跟有无此路径文件无关)  
# a = os.path.split(r'C:/Users/zhaoshuai/python\_学习/day4.py')  
# print(a)  
#将路径与后缀名分隔开  
# a = os.path.splitext(r'C:/Users/zhaoshuai/python\_学习/day4.py')  
# print(a)*

*#筛选后缀名是.py的文件  
# print(os.listdir())  
# for i in os.listdir():  
# if os.path.splitext(i)[1]=='.py':  
# print(i)  
# a = [i for i in os.listdir() if os.path.splitext(i)[1]=='.py']  
# print(a)*

*#创建10个目录 目录名user1到user10***for** i **in** range(1,11)**:** os.mkdir(**'user{}'**.format(i))  
*#删除目录***for** i **in** range(1,11)**:** os.rmdir(**'user{}'**.format(i))

python中的正则表达式 re模块 python自带的

筛选内容（筛选数据）

*正则表达式 re模块  
# import re  
# 导入re模块  
# with open('b.txt','r',encoding='utf-8') as f:  
# a = f.read()  
# print(a)  
# compile 编译 将筛选条件编译成计算机识别的正则  
# patt = re.compile('s(.\*)g') #只想要s和g中间的内容 加上括号  
# .匹配除了换行符之外的任意字符  
# 给.加功能 re.S 让.匹配任意字符  
# re.I 让正则表达式不区分大小写  
# findall 匹配所有符合条件的数据 # findall 编译的功能 想要匹配的内容直接写在finadll里面  
# 贪婪模式： 尽可能多的去匹配  
# 非贪婪模式：尽可能少的去匹配 ? 加?表示非贪婪模式  
# c = patt.findall(a)  
# print(c)  
# sub 替换 根据正则表达式替换字符串的内容  
# c = re.sub('[0-9]+','中国',a) #第一个是替换的条件 第二个是替换成的内容 第三个是在哪个地方替换  
# [0-9]+ 表示0-9的所有数字  
  
# print(c)*