Python

1. 简介

Python 是一门解释型的，面向对象的高级程序设计语言，基于c语言开发的

能做什么

人工智能

自动化测试

爬虫

数据分析

网站后台程序

优点

1. 简单2、开源免费 3、提供了完善的代码库

缺点

1. 代码不能加密 2、运行速度慢（相对的）

#!/bin/python 定义解释器

#-\*- coding:utf-8 -\*- 定义编码方式

Print 输入 表式需要从键盘输入

格式： 变量名=input(‘提示语’)

a=input(‘请输入用户名’)

Print=(a)

Print 输出 python print必须加括号

如果要打英语或字符必须加引导

Pint(‘河南’)

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  
 2、不能和python中的函数相同

打印变量名时不需要加$

Print(a)

变量的值如果是字符必须加引导

多变量赋值：变量名1，变量名2……=值1，值2……

a,b,c=1,2,3

python 解释器

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

jpython：java语言开发的；将python代码解释成java的字节码

pypython：python语言开发；

作用；提高了python的执行速度

数据类型

整数、空值、浮点数、布尔值、集合、列表、元组、字典

字符串

整数(int)：定义整数

1. 变量名=值（整数） 2、变量名=int（值）

第一个python的内置函数(type(变量名))：查看变量的数据类型

例1：abc=123

Print(type(abc))

例2：bc=12.6

abc=int(bc) : int将其他的数据类型更改为整数

print(pyte(abc))

浮点数（float）

定义浮点数 1、变量名=值（小数） 2、变量名=float（值）

例： abc=12

abc=float(abc）: 将其他的数据类型更改为小数

空值 None

布尔值(bool) True(真) False(假)

判断 12==‘ab’ 双等于

abc=12<11

print=(abc) 结果为假 12>11结果为真

字符串(str) 一串字符的集合

1. 变量名=’值’(任意) （2）变量名=str(‘值’)

注：字符串不能与数字相加

特点：1、不可变 2、支持索引 3、支持切片

1. 根据下标位置取对应的原索叫索引

格式：变量名[下标位]

3、切片： 格式：变量名[下标值1：下标值2]

取值范围：从1开始到2结束，不取2上面的元素

格式：1、变量名[：2]冒号在后代表从最后开始取值，冒号 在前代表从最前开始

1. 变量名：[下标值1：：]

取值范围 ：：在后表示最后，

：：在前表示以这个数字大 小分组取组中的第一

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( )

例：abc=’qwer’

f=abc.upper( )

print(abc)

print(f)

2、将字符串的首字母变大写

f=abc.capitalize( )

print（f）

3、将字符串大写变小写，小写变大写

f=abc.swapcase( )

print(f)

4、统计字符串的某个元素有多少个

abc=’qwertyui’

f=abc.count(‘e’)

print(f)

5、获取某元素的下标号

abc=’qwertyui’

f=abc.index(‘w’)

print(f)

6、字符串中的某一个元素替换

abc=’qwertyqu’

f=abc.replace(‘q’,’789’)

f=abc.replace(‘q’,’789’,2)选择替换多少个

print(f)

7、判断字符串是否以某个元素开头

abc.”qwertyui”

f=abc.startswith(‘qw’)

Print(f)

8、判断字符串是否以某个元素结尾

f=abc.endswith(‘q’)

Print(f)

9、填充字符串（格式化字符）

1》用大括号在字符串中占位置

abc=”1+{ }={ }” 换位 abc=’1+{小红}={小明}’

f=abc.format(4,5) f=abc.format(小明=3，小红=2)

print(f)

2》用%s在字符串中占位置

abc=‘1+%s=%s’

f=abc %(123,’qwer’)

print(f)

%s能填充任意字符 %d只能填充整数

10、去除字符串的空格

abc=‘qwertyui’

f=abc.strip( )

print(f)

f=abc.lstrip( )去除左边空格

f=abc.rstripc( )去除右边空格

11、将字符串变成列表

以某个元素为分割符，将字符串变成列表

abc.=’qwertyu’

f=abc.split(‘g’)

print(f) [‘e’,’rtyu’]

第二个python的内置函数(len(变量名))：统计变量名有多少个元素 例: abc=’fabcd7db’

f=len(abc)

print(f)

列表（list）一组数据的集合（等同于shell上的数组）

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)

Abc=[12,3,6,’ab’,’哈’] 注：值是有序列的

Print（type（abc））

特点：1、可变的 2、支持索引 3、支持切片

列表函数 abc=[12,”qwe”,’q@$#’]

1、给列表中添加新的（元素）数据（只能添加一个，默认最后一个） abc.append(10)

Print(abc)

2、添加到列表中任意位置（只能添加一个）

abc.insert(1,212)第一个是添加位置，第二是添加数据

1. 删除某个元素（只能删除一个）

abc.remove(12)

Print(abc)

1. 删除某个元素（通过下标位置）

abc.pop(-1)

Print（abc）

1. 排序（按列表升序排列）

注：列表中的所有元素是同一种数据类型

abc.=[12,13,1,2,3]

abc.sort( )

Print(abc)

1. 反转（将列表从后往前显示）

abc.reverse( )

print(abc)

1. 统计列表中的某个元素有多少个

f=abc.count(12)

print(abc)

1. 获取列表中某个元素的下标

f=abc.index(1)

print(f)

9、更新（将另外一个列表中的所有元素添加到本列表中）  
 abc=[1,2,3,4]

b=[‘12’,’13’,’abc’]

abc.extend(b)

print(abc)

1. 复制
2. 浅复制

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

bc=abc.copy( )

print(abc)

print(bc)

1. 深复制

import copy

abc=[1,2,[‘qwe’,’rty’],10]

bc=copy.deepcopy(abc)

abc[2].append(100)

print（abc）

print（bc）

元组 tuple 一组数据的集合（等同于shell上的数组）

格式：以小括号为表示，每个元素之间用逗号隔开

定义列表：（1）变了名=（‘值’或 值）（2）变量名=tuple（‘值’）

注：值是有序排列的

特点：1、不可变的 2、支持索引 3、支持切片

abc=(12,3,4)

abc(4,)

print(type(abc))

1. 统计元组中某一个元素有多少

abc=(12,’zxc’,3)

f=abc.aount(12)

print(f)

1. 获取元组中某个元组的下标

f=abc.index(12)

print(f)

字典 （dict）：以大括号为表示，元素是键值对的方式（key=value）

格式：变量名={key：value}

abc={‘name’：‘小明’，‘age’：‘19’，‘sex’：‘男’}

特点：1、可变的 2、无序的 3、不支持索引 4、支持字段获取值print（abc[‘sex’]）

注：字典中的key（键）必须是唯一的

字典的函数

1. 添加（向字典中添加键值对）

abc[‘age’]=123

print(abc)

1. 删除

f=abc.pop(‘sex’)

print(abc)

print(f)

pop本身保存删除的信息 ，abc.Popitem( )默认删除最后一个

1. 获取所有的键或者所有的值，获得所有的键值对

print（abc.keys( ) ） 键

print（abc.valuese( )） 值

print（abc.Items( )） 键值对

1. 更新 （将bc中所有的键值对，更新到abc字典中）

abc={‘name’：‘小明’，‘age’：‘19’，‘sex’：‘男’}

bc={‘abc’：123，‘des’：45}

abc.update(bc)

print(abc)

集合（set）：以大括号为标识，中间用逗号隔开

Abc={12,3,4,5,6}

Print（type（abc））

定义集合（1）变量名={值1，值2……}

（2）变量名=set（值） 注：值只有一个元素

如果定义一个空集合，不能使用大括号，只能用a=set（）

特点：1、无序的 2、不可重复的 3、不支持索引 4、可变的

集合的函数

1、添加 abc={12,23,34,67}

abc.add（110）

print(abc)

删除

运算符

算数运算符

+（加） 、 -（减）、\*（乘）、/（除）、//（整除）、%（求余）

比较运算符

>、 < 、>= 、<=、 !=、 ==（等于）一个等于是赋值， 两个等于是判断

逻辑运算符

与and 或or 非not

赋值运算符

+=(加)、 -=(减)、 \*=(乘)、 /=(除) 、//=(整除)、 %=（求余）

列 a=3

a+1 等于 a=a+1

Print（a）

成员运算符

in （在……里面） not in （在……里面）

判断语句

判断语句分为：单分支、双分支、多分支、嵌套

单分支： if 条件:

执行语句块

例：a=4

If a>3:

Print(‘hello’)

双分支： if 条件:

语句

else:

语句

例： a=2

if a>3:

print(‘hello’)

else:

print(‘hai’)

多分支： if 条件：

执行语句

Elif 条件：

执行条件

Elif 条件：

执行条件

Else：

执行语句

例：a=input（‘输入’）

a=int（a）

if a>90:

print(‘优秀’)

elif a>=80 and a<=90:

print(‘良好’)

elif a>=70 and a<80:

print(‘一般’)

else:

a<70

嵌套判断 if 条件：

If 条件：

1、条件后面要加冒号 2、执行语句块前面要加缩进

Python

循环语句 （for，while）

For格式：for 变量名 in 范围

执行语句块

范围：指的是一组数据的集合（列表，元组，字符串，字典）

函数 range（范围）

range（1,2,3）

1. 括号中有一个数时，表示默认从0开始到此数字结束取不到此数字
2. 括号中有两个数时，表示从第一个数字开始到第二个数字结束，取不到第二个数字
3. 括号中有三个数时，第3个数表示递进

循环中嵌套判断

Break 终止循环

Continue 跳过本次循环

例1： a=[1,2,3,4,5]

for i in a:

if i==3:

breake或continue

print(i)

例2：for i in 范围：

if 条件

执行语句块

elif 条件：

执行语句块

else：

执行语句块

import random

a=random.randint(1,10)

（从1-10随机选中一个数，赋值给a）

循环嵌套循环

For语句中加入for语句：大循环执行一次，小循环执行一轮。

例： for i in range(3):

print(‘#########’)

for j in range(3):

print(‘hello’)

for……else……语句

原理：只要循环没有被brak掉，就执行else语句

a=input(‘输入’)

b=len(a)//2

for I in range(b):

if a[i] != a[-i-1]:

print(‘不是回文’)

break

else：

print（‘是回文’）

enumerate（列表or元组 or字符串）

作用：将下标位置与数据一一对应

例： a=‘abcdefg’

for i in enumerate（a）：

print（i）

循环语句 while循环

格式：while 条件：

执行语句

While 适合场景：1、无线循环

For 适用场景：通常用来循环有序列的（下表位）数据时

循环嵌套 1、while……while……

while……for……

while……if……

只要循环没有被break掉，就执行else语句

报错：三部分

1行 报错的文件，报错行数

2行 报错的语句

3行报错的类型和描述

TypeError：1、不同的数据类型作运算 2、数据类型不能更改

NameError：（命名错误）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

例 try：

a=‘123’

Print（a+1）

Except：

Print（123）

例 try：

a=‘123’

print(a+1)

except Exceptio as n： as：重命名赋值

print（n） sum：求和

默认检查到所有的报错（Exceptio） max：最大

try： min：最小

a=‘123’

Print(a+1)

Except Typerror：

print（‘hello’）

try：

a=‘123’

print（a+b）

Except Typerror：

Print（‘hello’）

Except后跟某一类型的报错，只检测这一类型的错误

Try……except……else

原理：只有try语句没有错误，才执行else语句

Try……except……finally……

原理：无论执行try还是except，都执行finally

Raise自定义异常（触发异常）通常用于开发

异常处理语句

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；该中断而没有中断

格式：try……except……语句

例： try：

a=‘123’

print(a+1)

except:

print(‘hello’)

except：默认检测到所有的报错（Exception）

原理：将可能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