**目录**

[**May 2，2018** 1](#_Toc513812275)

[1、python的数字类型： 1](#_Toc513812276)

[2、算数运算符 1](#_Toc513812277)

[3、比较运算符 1](#_Toc513812278)

[4、变量 1](#_Toc513812279)

[5、对象 1](#_Toc513812280)

[6、内建函数 1](#_Toc513812281)

[**May 3, 2018** 2](#_Toc513812282)

[1、pow函数 2](#_Toc513812283)

[2、换行： 2](#_Toc513812284)

[3、基本输入、输出函数 2](#_Toc513812285)

[4、if语句 3](#_Toc513812286)

[5、if语句的真值表达式 5](#_Toc513812287)

[6、if语句嵌套 5](#_Toc513812288)

[7、条件表达式 7](#_Toc513812289)

[8、pass语句 7](#_Toc513812290)

[9、布尔运算 8](#_Toc513812291)

[10、布尔与操作 and 8](#_Toc513812292)

[11、布尔或操作or 8](#_Toc513812293)

[12、正负号运算符 9](#_Toc513812294)

[练习： 9](#_Toc513812295)

[**May4,2018** 13](#_Toc513812296)

[1.字符串 13](#_Toc513812297)

[2 用转义序列代表特殊字符 13](#_Toc513812298)

[3 raw字符串（原始字符串） 14](#_Toc513812299)

[4 字符串的运算 14](#_Toc513812300)

[5 字符串的比较运算 15](#_Toc513812301)

[**6 in/not in 运算符** 15](#_Toc513812302)

[7 字符串的索引操作 16](#_Toc513812303)

[**8 切片** 16](#_Toc513812304)

[9 python3中常用的序列函数： 18](#_Toc513812305)

[10字符串编码转换 18](#_Toc513812306)

[11 正数转换为字符串函数 19](#_Toc513812307)

[12 字符串的构造函数str 19](#_Toc513812308)

[May5（周串讲） 22](#_Toc513812309)

[May7 24](#_Toc513812310)

[1 常见字符串方法： 24](#_Toc513812311)

[2 字符串的格式化表达式 26](#_Toc513812312)

[3 占位符和类型码之间的格式语法 26](#_Toc513812313)

[4 循环语句 28](#_Toc513812314)

[4.1 while语句语法说明： 30](#_Toc513812315)

[4.2 while注意事项： 30](#_Toc513812316)

[5 while语句嵌套 31](#_Toc513812317)

[6 break语句 33](#_Toc513812318)

[7 死循环 33](#_Toc513812319)

[**May8** 36](#_Toc513812320)

[1 for语句 36](#_Toc513812321)

[1.1 作用： 36](#_Toc513812322)

[1.2 for 语法说明： 37](#_Toc513812323)

[2 range函数 37](#_Toc513812324)

[3 for语句的嵌套 38](#_Toc513812325)

[4 continue语句 40](#_Toc513812326)

[5循环总结： 41](#_Toc513812327)

[6 列表list 42](#_Toc513812328)

[6.3 列表的比较运算 43](#_Toc513812329)

[May9 45](#_Toc513812330)

[1 列表的 in / not in 45](#_Toc513812331)

[2 列表的 索引index / 切片 slice 45](#_Toc513812332)

[3 列表的索引赋值 45](#_Toc513812333)

[4 列表的切片 45](#_Toc513812334)

[4.1 切片步长不为1的切片赋值 46](#_Toc513812335)

[5 del语句用于删除列表元素 47](#_Toc513812336)

[6 Python3中常用的列表方法（method) 49](#_Toc513812337)

[7 字符串文本解析方法split 和 join 50](#_Toc513812338)

[8 深拷贝deep copy 和浅拷贝shallow copy 51](#_Toc513812339)

[9 列表推导式list comprehensive 51](#_Toc513812340)

[9.1列表推导式的嵌套 52](#_Toc513812341)

[May10 54](#_Toc513812342)

[1 元组tuple 54](#_Toc513812343)

[2 字典dict 55](#_Toc513812344)

[2.1字典的构造函数dict 56](#_Toc513812345)

[2.2 不可变类型 56](#_Toc513812346)

[2.3 可变类型 56](#_Toc513812347)

[2.4 字典的键索引 56](#_Toc513812348)

[2.5添加/修改字典元素 57](#_Toc513812349)

[2.6 字典中的 in/not in运算符 57](#_Toc513812350)

[2.7 可以用于字典的内建函数 58](#_Toc513812351)

[2.8 字典推导式 60](#_Toc513812352)

[May11 63](#_Toc513812353)

[1 集合set 63](#_Toc513812354)

[1.1定义： 63](#_Toc513812355)

[1.2集合的创建 63](#_Toc513812356)

[2 集合的构造函数set 63](#_Toc513812357)

[3 集合的运算 63](#_Toc513812358)

[4 python3中可以用于集合的函数 64](#_Toc513812359)

[5 集合的方法 65](#_Toc513812360)

[6 集合推导式 67](#_Toc513812361)

**May 2，2018**

1、python的数字类型：

整形：int； 浮点型：float 布尔型bool； 复数类型：complex；

空值：None;

Eg：int（值）

2、算数运算符

+ - \* / //（地板除） %（取余数） \*\*

3、比较运算符

> >= < <= ==（等于） !=（不等于）

4、变量

绑定数据对象，在辅助语句下“变量 = 表达式”（创建变量或改变变量）

符合赋值运算符

+= - = \*= /= \*\*=

Eg:a+=2相当于a=a+2

Is / is not

5、对象

Id(obj)返回对象的ID地址

Type（obj）返回对象的类型

del 语句（删除变量） eg：del变量名

6、内建函数

Round(数值，取精确位数) 四舍五入，但是计算机以2进制计算。

Complex（） 转换为复数。

7、换行符：\ （反斜杠）

**May 3, 2018**

Python函数：

1、pow函数

pow（x , y , z=None） 两个参数时相当于x\*\*y；三个参数时，相当于x\*\*y%z

eg： >>>pow(2,3) # 等同于2的3次方

>>>pow(2,3,5) #等同于2的3次方的得数，对5求余数。

2、换行：

显示换行： 折行号 \

Eg： i = 1 + 2 + 3 \

+ 4 + 5

Print（i） #用“\”显示换行

隐藏换行：所有的括号内换行 “( ) [ ] { }”三种

Eg：i = (1 + 2 + 3

+4 + 5)

3、基本输入、输出函数

①基本输入函数 input

作用：从标准输入设置上读取一个字符串（末尾的换行符会被删除）

格式：input（“提示字符串”）

说明：返回输入的字符串（仅限python3）“提示字符串”可以为空

Eg：s = input("请输入字符串：")

print("您输入的字符串是：",s)

②基本输出函数print

作用：将一系列的值以字符串形式输出到标准输出设备上，默认为终端

格式：

Print（value,…，sep=’ ’，end=’\n’） end=’ ’ 不换行

选项关键字参数为：

Sep两个值之间的分隔符，默认为一个空格（缺省参数）

End输出完毕后再流末尾自动追加一个字符，默认换行符 ’ \n’

Eg：print('===以下是给定sep="#"的打印方式')

print(1, 2, 3, 4, sep="#")

换行

print('以下关键字参数end="\n\n\n\n"')

print(1,2,3,4,end="\n\n\n\n")

print("我是程序的最后一句")

eg：分别输入当前的小时，分钟，秒，计算当前时间距离00:00:00过了多少秒？

s = input("请输小时：")

hour = int(s)

s = input("请输入分钟：")

minute = int(s)

s = input("请输入秒：")

second =int(s)

second = hour\*60\*60+minute\*60+second

print("距离00：00：00已经过了",second,"秒")

4、if语句

作用：

让程序根据条件选择性的执行某条语句过着某些语句。

说明：

if语句又叫条件语句，也叫分支语句。

语法：

if 真值表达式1：

语句块1

elif 真值表达式2：

语句块2

elif 真值表达式3：

语句块3

……

else：

语句块

语法说明：

elif 子句可以有0个，1个或者多个

else 子句可以有0个或者1个且具有能放在此if 语句的最后

要求：

if 语句内部的语句常要以4个空格的锁紧来表示包含关系，相同的缩进格式代表相同的所属级别。

Eg1：判断输入的数是奇数还是偶数

s = input("输入一个数：")

n = int(s)

if n % 2 ==0:

print("您输入的是偶数")

else:

print("您输入的是奇数")

eg2：任意输入一个数：1、判断这个数是否大于100；2、判断这个数是否小于0；3、判断这个数是否在20~50之间

s = input("输入一个数字：")

n =int(s)

if n>100 :

print("大于100")

elif n<0 :

print("此数小于0")

elif 20<n<50 :

print("在20~50之间")

else:

print("其他")

eg3:输入一个数字，判断这个数字是0，还是正数或者负数。

n = int(input("输入一个数字："))

if n == 0:

print("您输入的是0")

elif n > 0 :

print("您输入的是正数")

else:

print("您输入的是负数")

eg4：输入一个季度1~4输出这个季度有哪里个月，如果输入不是1~4的数，提示您输入的有误

n = int(input("输入一个季度（１～４）"))

if n==1 :

print("一月，二月，三月")

elif n==2 :

print("四月，五月，六月")

elif n==3 :

print("七月，八月，九月")

elif n==4 :

print("十月，十一月，十二月")

else :

print("您输入的有误")

eg5：输入一年中的月份（1~12），输入这个月在那个季度，如果输入的是其他数，提示您输入的有误！

n = int(input("输入月份"))

if 1<=n<=3:

print("春季")

elif 4<=n<=6:

print("夏季")

elif 7<=n<=9:

print("秋季")

elif 10<=n<=12:

print("冬季")

else:

print("您输入的有误")

5、if语句的真值表达式

if 100:

print(“真值”)

#等同于

if bool(100):

print(“真值”)

bool（x）返回假的情况：

x为：

0,0.0,0+0j，False，None，” ”（空字符串）

[]空集合

{}空字典

Set()空列表

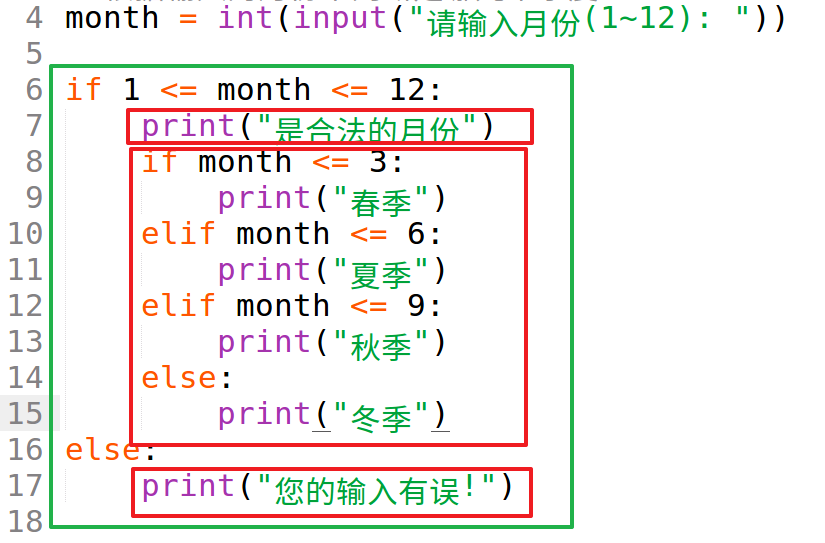
()空元祖

6、if语句嵌套

if语句本身是由多条句子组成的一条复合型语句

if语句可以作为语句嵌套到另一个语句的内部

eg：根据输入的月份来判断是哪个季度



练习：输入一个学生成绩（0~100），判断这个学生的成绩是优秀90~100，良好80~89，及格60~69，不及格，成绩不合法5种状态。

a = int(input("输入一个学生的成绩（0~100）"))

if 0<=a<=100 :

print("合法成绩")

if 90<=a<=100 :

print("优")

elif 80<=a<=89 :

print("良")

elif 60<=a<=69 :

print("及格")

else:

print("不及格")

else :

print("成绩不合法")

7、条件表达式

表达式一定能返回一个对象

语法：表达式1 if 真值表达式 else 表达式 2

（该表达式顺序执行，若if为真那么久直接送入表达式1，若为假则进入else直接输出）

作用：如果真值表达式的布尔环境值为Ture，则执行表达式1并返回结果的引用，否则执行表达式2并返回结果的引用。

Eg：商场促销，过100返20

money = int(input("请输入商品总额："))

pay =money - 20 if money >100 else money

print("您需要支付",pay,"元")

练习：

输入一个整数，1）用if语句计算出绝对值并且打印。

n = int(input("输入一个数："))

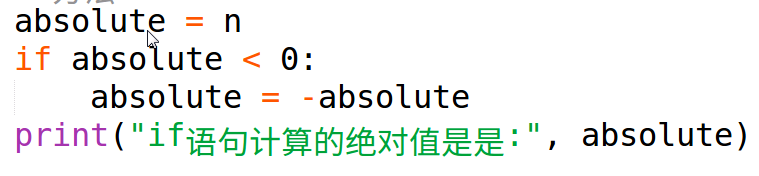
if n>=0 :

print(n)

else:

s = n\*(-1)

print(s)



2）用条件表达式计算出这个数的绝对值并且打印出来。

s = int(input("输入一个数"))

absolute = s\*(-1) if s<0 else s

print(absolute)

8、pass语句

作用：通常用来填充语法空白，Pass语句又名空语句。

语法：pass

Eg：判断一个学生成绩是否不合法。如果不合法给出警告信息。若合法什么都不做。

score = int(input("请输入成绩："))

if 0<=score<=100:

pass （#若没有pass程序缩进就会报错，因此需要空白语句占位）

else :

print("您的输入有误")

9、布尔运算

运算符： not and or

布尔非操作 not

语法：not x

作用：对x进行布尔取值，如bool（x）为True，则返回False，否则返回True

Eg：not True #False

Not False #True

Not 100 #False

10、布尔与操作 and

语法： x and y 注：x，y代表表达式

作用：优先返回假值对象，当bool（x）的值为False时，返回x，否则返回y

eg：True and True #True

True and False #False

False and True #False

False and False #False

(全真才为真，否则都为假，有假优先返回假)

11、布尔或操作or

语法：x or y

作用：优先返回真值对象，当x为True时返回x，否则返回y

eg：True or True #True

True or False #True

False or True #True

False or False #False

100 or 200 #100

100 or 0.0 #100

0 or 0.0 #0.0

0 or 200 #200

思考：

score = input("请输入成绩：")or'0'

score = int(score)

if score<0 or score>100:

print("成绩不合格")

else:

print("您的成绩",score)

12、正负号运算符

+ -

语法：+表达式

-表达式

eg：a = 5 b = -a c = +a

练习：

1. 北京出租车计费：

收费标准：3公里内收费13元，超过3公里后基本单价为2.3元/公里。空驶费：超过15公里后，每公里加收基本单价的50%作为返程的费用3.45元/公里

要求：输入公里数，打印出费用的金额（以元四舍五入）

kilometre = int(input("乘坐公里数"))

c = (kilometre-3)\*2.3+13

a = (kilometre-15)\*3.45+c

if kilometre <= 3 :

print("车费",13,"元")

elif 3 < kilometre <=15 :

print("车费",round(c),"元")

else :

print("车费",round(a),"元")

#方法2

km = int(input("乘坐公里数"))

fee = 13

if km > 3:

fee += 2.3 \* (km - 3)

if km > 15:

fee += 1.15 \*(km - 15)

print("车费",round(fee),"元")

1. 输一个学生的三科成绩：
2. 打印出最高分是多少分；2）打印出最低分是多少分；3）打印出平均分是多少分

score1 = int(input("输入学生成绩："))

score2 = int(input("输入学生成绩："))

score3 = int(input("输入学生成绩："))

a = (score1+score2+score3)/3

print("平均成绩",a)

if score1>=score2 and score1>=score3 :

print("最高成绩为",score1)

elif score2>=score1 and score2>=score3 :

print("最高成绩为",score2)

else:

print("最高成绩为",score3)

if score1<=score2 and score1<=score3:

print("最低成绩为",score1)

elif score2<=score1 and score2<=score3 :

print("最低成绩为",score2)

else:

print("最低成绩为",score3)

#方法2：

s1 = int(input("输入学生成绩："))

s2 = int(input("输入学生成绩："))

s3 = int(input("输入学生成绩："))

if s1 > s2 :

if s1 > s3:

print()

else:

print()

else:

if s2 > s3

print()

else:

print()

#经典：

s1 = int(input("输入学生成绩："))

s2 = int(input("输入学生成绩："))

s3 = int(input("输入学生成绩："))

m = s1

if s2 > m :

m = s2

if s3 > m:

m = s3

print("最高成绩",m)

1. 给出一个年份，判断是否为闰年并打印结果。闰年规则：每四年为一闰，每百年不闰，四百年又是一个闰年。例如：2016年，闰年 2100年平年 2400年是闰年。能被4整除的，但不能被100整除的是闰年。能被100整除的，又能被400整除的是闰年。

year = int(input("输入年份："))

a = year%4

b = year%100

c = year%400

if a==0 and b!=0 :

print("闰年")

elif b == 0 and c == 0:

print("闰年")

else :

print("平年")

1. BMI指数（body mass index）以称身体质量指数。BMI公式：

BMI=体重（kg）/身高的平方（m）

例如：一个69kg的人，身高是173cm BMI=69/1.73\*\*2=23.05

标准表：BMI<18.5体重过轻

18.5<=BMI<=24正常范围

BMI>24 体重过重

输入身高和体重，打印出BMI的值，并打印体重状况。

height = float(input("身高(m):"))

weight = int(input("体重（kg）:"))

BMI = weight//(height\*\*2)

if BMI<18.5 :

print("体重过轻")

elif BMI>24 :

print("体重过重")

else :

print("正常范围")

**May4,2018**

1.字符串

str

作用：用来记录文本（文字）信息

表示方法：在非注释中，凡是用引号括起来的部分都是字符串

‘ 单引号

“ 双引号

‘ ‘ ‘ 三引号

“ “ “ 三双引号

空字符串的字面值表示方式

‘ ‘ “ “ ‘ ‘ ‘ ’’’ “ “ “ ”””

单引号和双引号的区别：

单引号内的双引号不算结束符

双引号内的单引号不算结束符

1.1三引号字符串

作用：三引号字符串的换行会自动转换为行符’\n’

三引号内可以包含单引号和双引号

eg：’welcome to Beijing\n I like python!\n I am studing!’

三单引写法s='''welcome to beijing

i like python!

i am studing!'''

2 用转义序列代表特殊字符

字符串字面值中，用字符反斜杠(\) 后跟一些字符代表一个字符

字符串中的转义字符表

\’ 代表一个单引号 \0 空字符

\” 一个双引号 \0oo oo为两位八进制表示的字符

\\ 代表一个反斜杠 \xXX XX为两位16进制表示的字符

\n 换行 \uXXX Unicode16的16进制表示的字符

\r 返回光标至行首

\f 换页 \uXXXXXXXX Unicode32的16进制表示的字符

\t 水平制表符

\v 垂直制表符

\b 倒退

2.1常用的ASCII编码：

|  |  |  |
| --- | --- | --- |
| 字符 | 十进制 | 十六进制 |
| ‘0’ | 48 | 0x30 |
| ‘A’ | 65 | 0x41 |
| ‘a’ | 97 | 0x61 |

3 raw字符串（原始字符串）

字面值格式：

r’字符串内容’

r”字符串内容”

r’’’字符串内容’’’

r”””字符串内容”””

作用： 让转义字符 \ 无效

示例：

>>> a = 'C:\newfile\test.py'

>>> print(a)

C:

ewfile est.py

>>> print(len(a))

16

>>> a = r'C:\newfile\test.py'

>>> print(a)

C:\newfile\test.py

>>> print(len(a))

18 #字符

4 字符串的运算

算数运算符： + += \* \*=

+加号运算符用于字符串的拼接

eg：x=’abcd’+ ’efg’

print(x) #abcdefg

x += ‘123’

print(x) #abcdefg123

\*运算符用于生成重复的字符串

x=’123’

y=x \*2 #y = ‘123123’

x \*=’3’ #x =’123 123 123’

练习：写一个程序，打印一个高度为4的矩形方框要求输入一个整数，此整除代表矩形的宽度，输出此矩形

如：

请输入宽度：10

打印如下

##########

# #

# #

##########

s = int(input("输入一个整数"))

print('#'\*s)

print('#'+' '\*(s-2)+'#')

print('#'+' '\*(s-2)+'#')

print('#'\*s)

5 字符串的比较运算

运算符： > >= < <= == !=

示例：

‘A’<’B’ #True

‘B’<’a’ #True

‘ABC’>’AB’ #True

‘AD’<’ABC’ #False

‘ABC’==’abc’ #False

**6 in/not in 运算符**

作用：

in用于序列、字典、集合中，用于判断某个值是否存在于容器中，如果存在则返回True，否则返回False

格式：

对象 in 容器

示例：

s = ‘welcome’ to tarena!

‘to’ in s #True

‘weimingze’ in s #False

7 字符串的索引操作

python字符串str是不可以改变的字符序列

索引语法index：

字符串[整数表达式]

说明：

python序列都可以用索引（index）来访问序列中的对象（元素）；

python序列的正向索引是从0开始的，第二个索引为1，最后一个索引为len（s）-1

python序列的反向索引是从-1开始的，-1代表最后一个，-2代表倒数第二个，第一个是-len（s）

示例：

s = ‘ABCDE’

print(s[0]) #A

print(s[1]) #B

print(s[4]) #E

**8 切片**

从字符串序列中取出相应的元素重新组成一个字符串序列

语法：

s[(开始索引b):(结束索引e)(:(步长s))] ()内可以省略

语法说明：

开始索引是切片开始切下的位置0代表第1个

结束索引是切片的终止索引（但不包含终止点）

步长是切片每次获取完当前元素后移动的方向和偏移量

1. 没有步长，相当于取值完成后向右移动一个索引位置（默认值1）
2. 当步长为正数时，取正向切片：

步长默认为1，开始索引默认为0，结束索引默认为len(s)

1. 当步长为负整数时，取反向切片：

反向切片默认的起始位置为最后一个元素，终止为止是第一个元素前一个位置

示例：s = ‘ABCDE’

a = s[1:4] #a-->’BCD’

a = s[:4] #a-->’ABCD’

a = s[2:] #a-->’CDE’

a = s[0:2] #a-->’AB’

a = s[1:1] #a-->’ ‘

a = s[3:1] #a-->’ ’

a = s[::2] #a-->’ACE’

a = s[2::2] #a-->’CE’

a = s[::-1] #a-->’EDCBA’

a = s[3::-2] #a-->’DB’

练习：

1. 输入一个字符串，打印如下内容
2. 打印这个字符串的第一个字符
3. 打印这个字符串最后一个::字符
4. 如果这个字符串长度是奇数，打印中间这个字符（求字符串长度的函数是：len(s)）

# s = str(input('输入一个字符串'))

# l = len(s)

# a = l%2

# print(s[0:1])

# print(s[l-1:l])

# if a != 0:

# print(s[(l//2):(l//2+1)])

# else:

# pass

s = input("请输入字符串：")

print("第一个字符是：",s[0])

print("第二个字符是：",s[-1])

length = len(s) #表示长度

if length %2 == 1 :

center\_index=length // 2

print("中间的字符是：",s[center\_index])

1. 输入任意一个字符串，判断这个字符串是否是回文，回文是指中心对称的文字。如：上海自来水来自海上 ABCCBA

s = str(input("输入一段字符"))

a = s[::]

b = s[::-1]

if a == b:

print("是回文")

else :

print("不是回文")

9 python3中常用的序列函数：

len(seq)返回序列的长度

max（x）返回序列的最大值元素

min（x）返回序列的最小值元素

示例：

s = “ABCD”

print(len(s)) #4

print(max(s)) #D

print(min(s)) #A

10字符串编码转换

ord(c)返回一个字符串的Unicode编码值

chr(i)返回i这个值所对应的字符

示例：

print(ord(‘A’)) #65o

print(ord(‘中’)) #20013

练习：

1. 写一个程序，输入一段字符串，如果字符串不为空，则把第一个字符的编码打印出来。

s = input("输入一段字符串：")

l = len(s)

if l>0 :

a = ord(s[0])

print(a)

1. 写一个程序，输入一个整数值（0~65535），打印出这个数值所对应的字符。

s = int(input("输入0~65535之间的数值"))

a = chr(s)

print(a)

11 正数转换为字符串函数

hex(i)将整数转换为16进制的字符串

oct(i)将整数转换为8进制字符串

bin(i)将整数转换为2进制字符串

12 字符串的构造函数str

str(obj=’’) 将对象转换为字符串

示例：

s=123

print(str(s)+’456’) #123456

练习：

1. 用字符串\*运算符打印三角形，要求输入一个整数，此整数代表三角形左侧的字节数。如：请输入离左侧的距离：3

则打印：

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

s = int(input("输入一个整数："))

print(' '\*(s+3),"\*")

print(' '\*(s+2),"\*\*\*")

print(' '\*(s+1),"\*\*\*\*\*")

print(' '\*(s),"\*\*\*\*\*\*\*")

1. 输入三行文字，让这三行文字在一个方框层居中显示。

如：输入（不要中文）：

hello tarena！

my name is weimingze

python

显示如下：

+---------------------------+

| hello tarena! |

| my name is weimingze |

| python |

+---------------------------+

s1 = str(input("第一行文字:"))

s2 = str(input("第二行文字:"))

s3 = str(input("第三行文字:"))

a1 = len(s1)

a2 = len(s2)

a3 = len(s3)

s = max(a1,a2,a3)

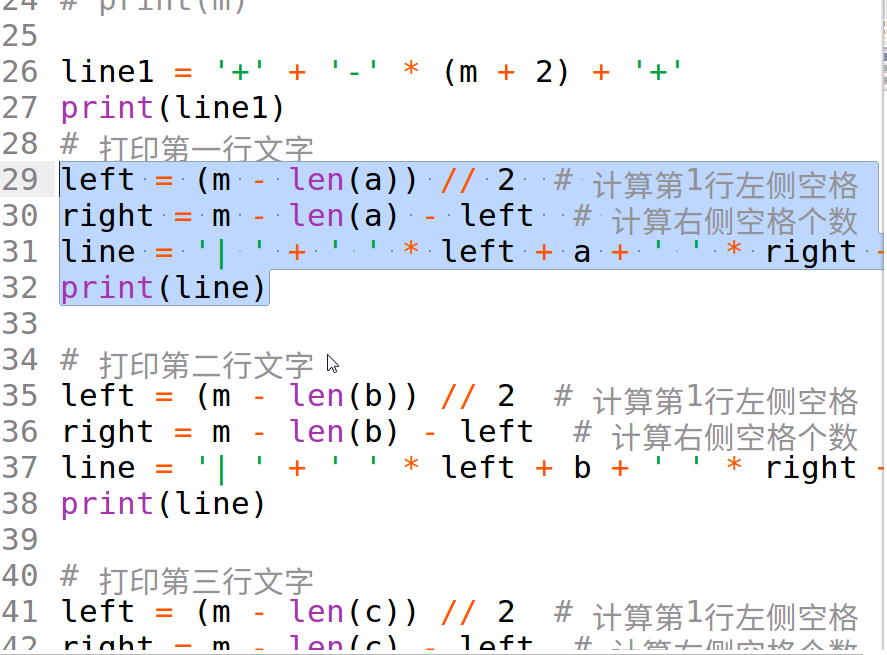
print('+'+'-'\*(s+2)+'+')

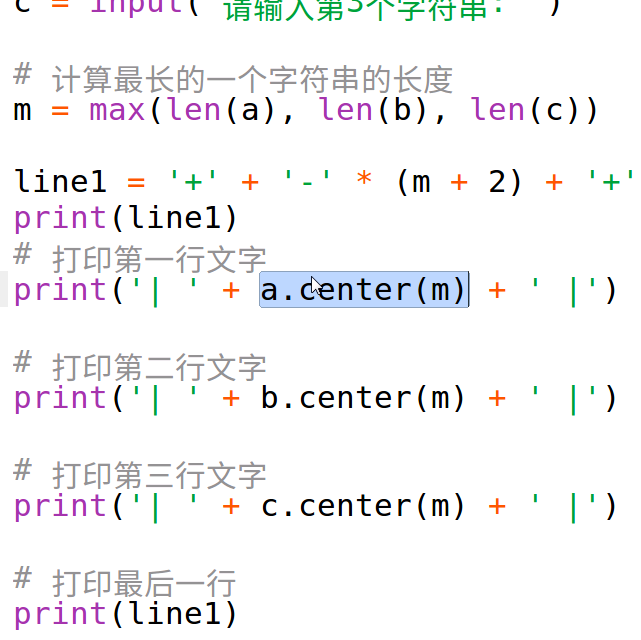
print('|'+(s+2-a1)//2\*' '+s1+(s+2-a1)//2\*' '+'|')

print('|'+(s+2-a2)//2\*' '+s2+(s+2-a2)//2\*' '+'|')

print('|'+(s+2-a3)//2\*' '+s3+(s+2-a3)//2\*' '+'|')

print('+'+'-'\*(s+2)+'+')





May5（周串讲）

1 虚拟控制台切换：

tt1,tt2,tt3,tt4,tt5,tt6,tt7

ctrl+alt+F1,F2,F3,F4,F5,F6,F7

2 命令

[用户名@计算机主机名 当前你所在目录]#

[用户名@计算机主机名 当前所在目录] $

#：管理员权限/root/ $:普通用户权限/home/

命令组成： 命令 [选项]……[参数1]……[参数2]

cat –n [参数]：在文档所有行之前表示行号

3 新建文档

1） touch / opt /a.txt

2） vim /opt /a.txt

用vim 创建步骤

1、vim a.txt 在目录下输入文件名为a的txt文件

2、i 输入i开始输入字符（插入模式）

字符例如：qweqwwaetgwe

4、ESC 退出文字输入模式，进入命令格式

5、末行模式（shift+:）:w(保存)q（退出） ！q（强制退出）

插入模式（i进入）-asdasdas(ESC)

命令模式：dd（删除）yy（复制）p（粘贴）

4 head，tail命令

作用：查看文档的部分内容

例如：

head -1 b.txt 查看文档b.txt第一行内容

head -2 b.txt 查看文档b.txt第一二行内容

tail -1 b.txt 查看文档b.txt最后一行内容

5 grep工具

作用：输出包含指定字符串的行

格式：grep[选项]…[参数（文档）]

grep wkl a.txt 查找wkl在文档a.txt中

6 ls查看命令

格式：ls[选项]……[目录]……

选项：

-l:以长格式显示

-d：以目录本身（不是内容）的属性

-h：提供容易读取的容量单位（K,M等）与-l一起使用

-a（All）：包括名称以“.”开头的隐藏文件

7 常用快捷键

|  |  |
| --- | --- |
| Ctrl + c | 结束当前进程 |
| Esc + . | 粘贴上一个命令的参数（我觉得吧） |
| Ctrl + l | 清屏 |
| Ctrl + u | 清空至行首 |
| Ctrl + w | 往回删除一个单词 |
| rm -rf | 强制删除 |
| cp -r | 递归复制 |

8 压缩与解压

tar [选项] [tar名称包] [被压缩文件1] [被压缩文件2]……

选项：

|  |  |
| --- | --- |
| -c | 创建压缩（创建归档） |
| -x | 释放归档 |
| -f | 制定归档文件的名称 |
| -z，-j，-J | .gz .bz2 .xz 三种格式 |
| -t | 显示归档文件中的文件清单 |
| -P | 保持归档内文件的绝对路径 |
| -C（大小写） | 指定释放路径 |

9 acl权限

getfacl：设置

setfacl：

May7

1 常见字符串方法：

见：str.html

字符串方法的调用语法

对象.方法名（方法传参）

注：方法的调用属于表达式，通常可以返回一个值或None

示例：

‘abc’.isalpha() #Ture

123.isalpha() （错误的语法）

| **方法** | **说明** |
| --- | --- |
| S.isdigit() | 判断字符串中的字符是否全为数字 |
| S.isalpha() | 判断字符串是否全为英文字母 |
| S.islower() | 判断字符串所有字符是否全为小写英文字母 |
| S.isupper() | 判断字符串所有字符是否全为大写英文字母 |
| S.isspace() | 判断字符串是否全为空白字符 |
| S.center(width[,fill]) | 将原字符串居中，左右默认填充空格 |
| S.count(sub[, start[,end]]) | 获取一个字符串中子串的个数 |
| S.find(sub[, start[,end]]) | 获取字符串中子串sub的索引,失败返回-1 |
| S.strip() | 返回去掉左右空白字符的字符串 |
| S.lstrip() | 返回去掉左侧空白字符的字符串 |
| S.rstrip() | 返回去掉右侧空白字符的字符串 |
| S.upper() | 生成将英文转换为大写的字符串 |
| S.lower() | 生成将英文转换为小写的字符串 |
| S.replace(old, new[, count]) | 将原字符串的old用new代替，生成一个新的字符串 |
| S.startswith(prefix[, start[, end]]) | 返回S是否是以prefix开头，如果以prefix开头返回True,否则返回False, |
| S.endswith(suffix[, start[, end]]) | 返回S是否是以suffix结尾，如果以suffix结尾返回True,否则返回False |
| 以下是不常用的 |  |
| S.title() | 生成每个英文单词的首字母大写字符串 |
| S.isnumeric() | 判断字符串是否全为数字字符 |

空白字符

是指空格,水平制表符(\t),换行符(\n)等不可见的字符

练习：输入一个字符串

1. 判断您输入的字符串有几个空格
2. 将原字符串的左右空白字符去掉，打印出有效字符的长度
3. 判断您输入的是否是数字

s = input("输入一段字符串：")

print(s.count(" "),"个空格")

s2 = s.strip()

print("左右空格去掉",len(s2),"个有效字符",s2)

if s2.isdigit():

print("有数字")

else:

print("无数字")

2 字符串的格式化表达式

作用：生成一定格式的字符串

预算符：%

语法格式：

格式字符串%参数值

格式字符串%（参数值1，参数值2，……）

说明：

格式化字符串中以%开头的为占位符，占位符的位置将用参数值替换

%右侧为参数值，当有多个参数值时用括号（）括起来，之间用逗号（，）隔开

示例：

fmt = ‘姓名’ %s,年龄：%d

name = ‘tarena’

age = 15

print（fmt % （name，age））

>>> fmt = "姓名:%s,年龄:%d"

>>> fmt % ("tarena",15)

'姓名:tarena,年龄:15'

>>> fmt = '班级:%s,姓名:%s'

>>> fmt % ("电子1621班","武凯伦")

'班级:电子1621班,姓名:武凯伦'

>>>

3 占位符和类型码之间的格式语法

% [格式语法] 类型码

格式语法：

* 左对齐

+ 显示正号

1. 补零

宽度（整数）

宽度.精度（整数）

示例：’%10d’ %123 #’ 123’

’%-10d’ %123 #’123’

’%10s’ %abc #’ abc’

’%-5s’ %abc #’ abc ’

’%05d’ %123 #’00123’

‘%7.3f’ %3.1415926 #’003.141’

| **占位符类型码** | **意义** |
| --- | --- |
| %s | 字符串 |
| %r | 字符串，使用repr 而不是str |
| %c | 整数转为单个字符 |
| %d | 十进制整数 |
| %o | 八进制整数 |
| %x | 十六进制整数(字符a-f小写) |
| %X | 十六进制整数(字符A-F大写) |
| %e | 指数型浮点数(e小写),如2.9e+10 |
| %E | 指数型浮点数(E大写),如2.9E+10 |
| %f, %F | 浮点十进制形式 |
| %g, %G | 十进制形式浮点或指数浮点自动转换 |
| %% | 等同于一个%字符 |

练习：

输入三行文字，让这些文字依次以20字符的宽度右对齐输出。

如：请输入第一行：hello world

请输入第二行：abcd

请输入第三行：a

输出结果为：

hello world

abcd

a

==================================================================

s1 = input("输入第一行字符：")

s2 = input("输入第二行字符：")

s3 = input("输入第三行字符：")

print("%10s" %s1)

print("%10s" %s2)

print("%10s" %s3)

思考题：以最长字符串为空格

s1 = input("输入第一行字符：")

s2 = input("输入第二行字符：")

s3 = input("输入第三行字符：")

a=max(s1,s2,s3)

b=len(a)

print(" " \* (b - len(s1)) + s1)

print(" " \* (b - len(s2)) + s2)

print(" " \* (b - len(s3)) + s3)

或

s1 = input("输入第一行字符：")

s2 = input("输入第二行字符：")

s3 = input("输入第三行字符：")

b=max(len(s1),len(s2),len(s3))

fmt = '%%%ds' % b

print('fmt = ', fmt)

print(fmt % s1)

print(fmt % s2)

print(fmt % s3)

4 循环语句

while语句与for语句

问题：输入一个数用n绑定，打印出n行的“hello world！”

print（“hello world！”）

while语句：

作用：根据一定条件，重复的执行一条语句或多条语句

语法：while真值表达式：

while真值表达式：

语句块

else：

语句块

示例：

i = 1#创建一个变量，用来控制whlie的循环

while i < 20 :#判断循环条件，如果为真则执行语句块

print("hello world!")

i += 1#修改循环变量

else:

print("条件不满足，循环结束！")

循环打印19条“hello world！”

练习：用while语句解决上述“问题”：

s =int(input("输入一个数："))

while s != 0:

print("hello world!")

s -= 1

else:

print("结束")

示例：

用while语句打印1~20的整数包含20

i = 1

while i <= 20:

print(i)

i += 1

练习：1输入一个整数，用end变量绑定，打印出1~end的所有整数，包含end

s = int(input("输入一个整数："))

i = 1

while i <= s:

print(i)

i += 1

2写程序，输入两个整数，第一个用begin绑定，第二个用end变量绑定，打印出begin~end的所有整数

s =int(input("输入一个整数"))

s2 = int(input("输入二个整数"))

while s<=s2:

print(s)

s=s+1

4.1 while语句语法说明：

1. 先执行真值表达式，判断True/False
2. 如果True则执行语句块1，然后跳转到第1步
3. 如果False则执行else子句部分语句块2，然后结束此while语句的执行；如果没有else子句，则直接结束while语句
4. else子句部分可以省略（同if语句类似）

4.2 while注意事项：

1. 要控制真值表达式访问死循环
2. 通常用真值表达式内的变量来控制循环条件
3. 通常要在循环语句块内改变循环变量来控制循环的次数和变量的走向

练习：

1. 打印1~20的整数，打印在一行内显示，每个数字之间用一个空格分隔开。

（print（…,end=” ”））

i = 1

while i<=20:

print(i,end=" ")

i+=1

else:

print()#换行

1. 打印1~20的整数，每行5个，打印四行。（while内可以嵌入if语句）

i = 1

while i<=20:

print(i,end=" ")

if i % 5 == 0:

print()

i+=1

1. 用while语句打印10~1的所有整数（包含1）

i = 10

while i > 0:

print(i)

i -= 1

练习：

1. 写程序，用while循环来计算1+2+3+4+……+99+100的和

i = 1

a = 0

while i<=100:

a += i

i += 1

else:

print(a)

1. 用while语句实现打印三角形，输入一个整数，分别表示三角形的宽度和高度，打印出相应的直角三角形。

\*

\*\*

\*\*\*\*

\*\*\*\*\*

w = int(input("输入一个整数："))

i = 1

while i<=w:

print("\*"\*i)

i += 1

5 while语句嵌套

while语句本身是语句，和其他语句一样，可以放在其他符合语句的内部

while嵌套示意：

while 真值表达式：

……

while 真值表达式2：

……

else：

……

……

else：

……

问题：打印1~20之间的整数一行1 2 3 4 5 6 ……19 20

打印多行

c = 1

while c<=10:

i = 1

while i<=20:

print(i,end=" ")

i += 1

else:

print()

c +=1

练习：

1. 输入一个数，打印指定宽度的正方形

如：输入5

打印如下正方形：

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

s = int(input("输入一个数字："))

a = 1

while a <= s:

i = 1

while i<=s:

print(i,end=" ")

i += 1

else:

print()

a += 1

else:

print()

6 break语句

作用：用于循环语句（while，for语句）中，用来终止当前循环语句的执行

break说明：

1.当break语句执行后，次循环语句break之后的语句将不在执行。

2.break语句通常和if语句组合使用。

3.break语句终止循环时，循环语句的else子句的语句将不会执行

4.break语句只能终止当前循环语句的执行，如果有循环嵌套时，不会跳出嵌套的外重循环。

5.break语句只能在循环语句（while或for语句）内部使用

7 死循环

死循环是指循环条件一直成立的循环，死循环常用break语句来终止；死循环的else子句永远不会执行

示例：

while True:

n = int(input(“请输入：”))

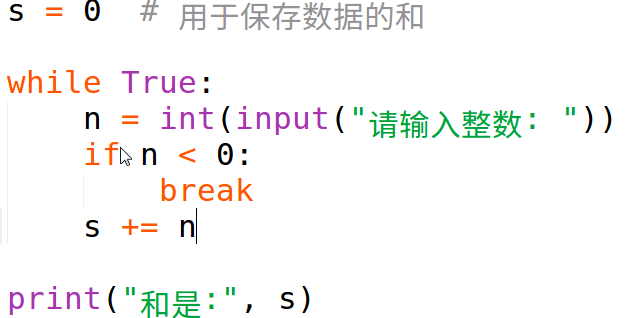
if n == 0:

break

print(n)

练习:

1. 任意输入一些整数，每次输入一个，当输入负数时结束输入，当输入完成后，打印这些数的和。



1. 写程序，用while实现打印三角形。

要求输入一个整数表示三角形的宽度和高度，打印出如下三种直角三角形

s = int(input("输入一个数表示三角形的高度和宽度："))

i = 1

while s>0:

print(" "\*(s-1) + "\*"\*i)

s -= 1

i += 1

\*

s = int(input("输入一个数表示三角形打高度和宽度："))

i = 0

while s>0:

print(" "\*i+"\*"\*s)

s -= 1

i += 1

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

\*\*\*\*

s = int(input("输入一个数表示三角形的高度和宽度："))

while s>0 :

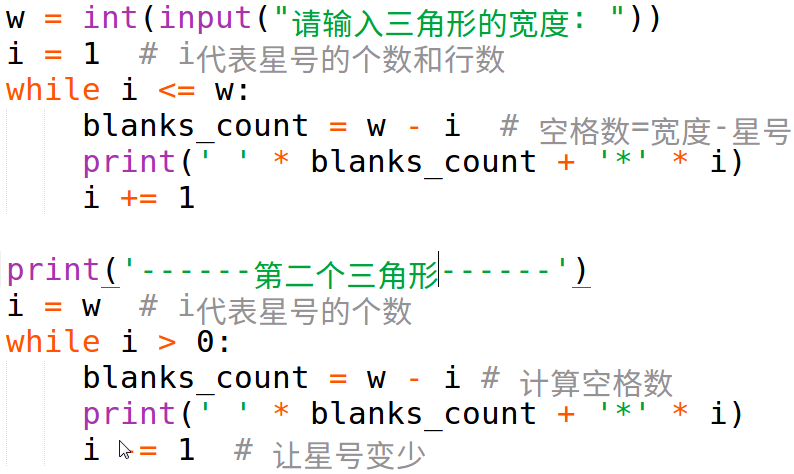
print("\*"\*s)

s -= 1

\*\*\*

\*\*

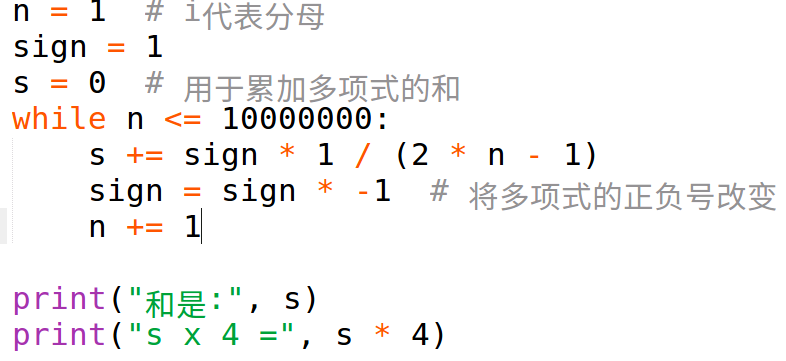
\*



3.写程序，求多项式的和：1/1 – 1/3 + 1/5 – 1/7 + 1/9……+1/（2\*n-1）

n取最大值1000000

1. 打印这个和
2. 打印这个和乘以4的值？



**May8**

1 for语句

1.1 作用：

用来遍历（每个数据都经历一次）可迭代对象的数据元素

可迭代对象是指能依次或缺数据元素的对象

可迭代对象包括：

字符串str

------以下之后再讲-------

列表list

元组tuple

字典dict

集合set

……

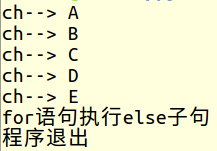
for 语句语法:

for 变量列表 in 可迭代对象：

语句块1

else：

语句块2



示例：

s = 'ABCDE'

for ch in s:

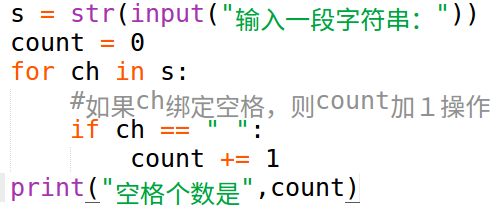
print('ch-->',ch)

else:

print("for语句执行else子句")

print("程序退出")

练习：任意输入一个字符串，判断这个字符串中有几个空格’ ’(要求不使用S.count方法)建议使用for语句实现



1.2 for 语法说明：

1. 可迭代对象每次提供一个元素依次赋值给变量列表中的变量，复制完毕后执行语句块1，重复执行此步骤，直到可迭代对象不能提供数据为止。
2. 可迭代对象提供完元素后，执行else子句部分的语句块2，然后退出此for语句
3. else子句可以省略（同while语句类似）
4. 当再循环内部用break终止循环时，else子句部分语句不会执行。

2 range函数

格式见：

help（range）

函数：

1.range（stop）从零开始，每次生成一个整数后加1操作，直到stop为止（不包含stop）

2.range(start,stop[,step])从start开始，每次生成一个整数后移动step，直到stop为止（不包含stop，且step可以是负数）

作用：

用来创建一个生成一系列整数的可迭代对象（也叫整数序列生成器）

说明：

range返回的对象是可迭代对象。可以用for语句中

示例：

range(4) #生成0,1,2,3

range(3,6) #生成3,4,5

range(1,10,2) #生成1,3,5,7,9

range(5,0,-2) #生成5,3,1

range(4,0) #None

练习：

1用for语句打印1~20的整数，打印在一行1 2 3 4 5 6 7 8 9……18 19 20

for x in range(1,21):

print(x,end=" ")

else:

print()

2求100以内有哪些整数与自身加1的成绩再对11求余结果等于8？

for x in range(1,101):

if (x+1)\*x%11 == 8:

print(x)

3计算1+3+5+7+……+99的和用while和for语句两种方法来实现

a = 0

for x in range(1,100,2):

a +=x

print(a)

a = 0

n = 1

while n<=100:

a += n

n += 2

else：

print(a)

3 for语句的嵌套

for语句内部可以放任意语句，包括for语句和while语句。

示例：

for x in “ABC”:

for y in ‘123’:

print(x+y)

思考：count的值是多少？

count = 0

for x in range(5):

for y in range(10):

count += 1

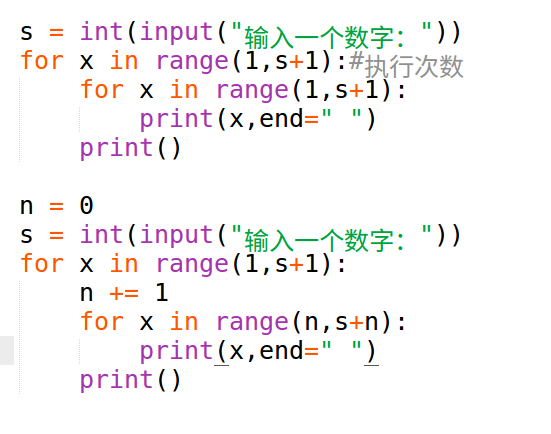
print(count) count=50

print(x ,y) x = 4 y = 9

练习：

1.写程序，输入一个整数n代表正方形的高度和宽度，打印数字组成正方形：如

输入5：



1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

2.在第一个题的基础上，

输入4：

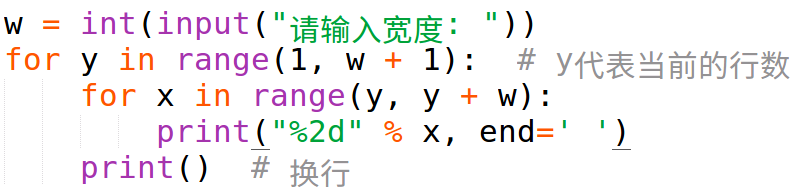
1 2 3 4

2 3 4 5

3 4 5 6

4 5 6 7

方法二：



4 continue语句

作用：

用于循环语句（while，for语句）中，不再执行本次循环内continue之后的语句，重新开始一次新的循环

说明：

1.在while语句中，执行continue语句将会直接跳转到while语句的真值表达式处重新判断循环条件。

2.在for语句中，执行continue语句，将会从可迭代对象中取下一个元素，绑定变量后再次进行循环

示例：

for x in range(5):

if x == 2:

continue

print(x)

练习：1输入一个整数用begin绑定，再输入一个整数用end绑定，打印出从begin~end（包含end）的所有偶数（建议用continue语句跳过奇数）

begin = int(input("输入第一个数："))

end = int(input("输入第二个数："))

for x in range(begin,end+1):

if x % 2 != 0:

continue

print(x)

2用while循环，实现打印10以内的偶数

用continue语句

i = 0

while i < 10:

if i % 2 == 1:

i += 1

continue

print(i)

i += 1

3练习：求1~100（包含100）之间所有不能被5,7,11整除的数是多少？

（建议用continue语句实现）

a = 0

for x in range(1,101):

if x % 5==0:

continue

elif x % 7==0:

continue

elif x % 11==0:

continue

a += x

print(a)

5循环总结：

while语句

for语句

-字符串

-range（）函数

break语句

continue语句

练习：

输入一个整数n，判断这个整数是否是素数(prime)素数（素数是只能被1和自身整除的数）如：2 3 5 7 11……

方法：

用排除法，一旦n能被2~n-1的数整除就不是素数，否则就一定是素数。

n = int(input("输入一个整数："))

if n<2:

print(n,"不是素数")

exit()

for x in range(2,n):

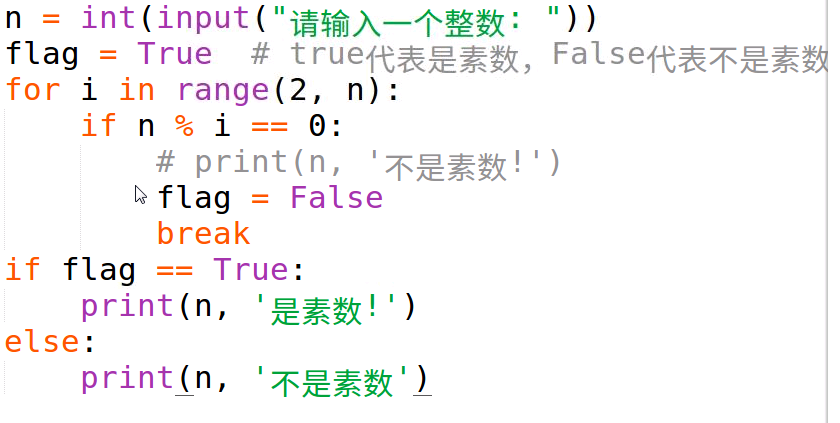
if n % x == 0:

print(n,"不是素数")

break

else:

print(n,"是素数")



6 列表list

列表是由一系列特定元素组成的，元素和元系之间没有任何关联关系，但他们之间有先后顺序关系。

列表是一种容器

列表是序列的一种

列表是可以被改变的序列

python中的序列类型简介（sequence）：

字符串str

列表list

元组tuple

字节串bytes

字节组bytearray

创建一个空列表的字面值

L = [ ] #L绑定空列表

创建非空列表：

L = [ 1,2,3,4]

L = [“北京”，”上海” , ”重庆”]

L =[1,’two’ , 3 , ’四’]

L =[1 , 2, [3.1, 3.2 ,3.3],4]

6.1列表的构造（创建）函数list

list（）生成一个空的列表等同于[ ]

list(iterable)可以用迭代对象创建一个列表

示例：

L = list( ) L为空列表，等同于L =[ ]

L = list(“ABCD”) #L->[‘A’ , ‘B’ , ‘C’ , ‘D’ ]

L = list(range(1,10,2))

6.2列表的运算

算术运算 + += \* \*=

+用于拼接列表

x=[1,2,3] y=[4,5,6] z = x + y #z = [1,2,3,4,5,6]

+=用于原列表与右侧可迭代对象进行拼接，生成新的列表

x = [1,2,3]

x += [4,5,6] # x=[1,2,3,4,5,6]

x = [1,2,3]

x +=’ABC’ # +=右侧必须是可迭代对象

\*生成重复列表

x = [1,2,3] \* 2 # x = [1,2,3,1,2,3]

\*=用于生成重复的列表，同时用变量绑定新列表

x = [1,2]

x \*=3 # x = [1,2,1,2,1,2]

6.3 列表的比较运算

运算符：

< <= > >= == ！=

x = [1,2,3] y = [1,2,3]

x != y #False

x > [1,2] #True

x < y #False

[1,3,2] > [1,2,3] #True

[‘AB’ , ‘CD’] > [‘AC’ , ‘BD’] #False

[1, ‘two’] > [‘two’ , 1 ] #TypeError

练习：

1.输入一个整数，代表树干的高度，打印一颗“圣诞树”

如：

high = int(input("输入一个整数代表树干的高度："))

for i in range(1,high+1):

s = (2\*i-1)\*"\*"

print(s.center(high\*2-1))

continue

c = 1

x = "\*"

while c<=i:

print(x.center(high\*2-1))

c += 1

输入：2

打印

\*

\* \* \*

\*

\*

输入：3

\*

\*\*\*

\*\*\*\*\*

\*

\*

\*

2.用循环语句生成如下字符串

‘ABC……XYZ’

‘AaBbCc……XxYyZz’

for i in range(65,91):

letter1 = chr(i)

letter2 = chr(i+32)

print(letter1+letter2,end=" ")

print()

提示：用ord和chr函数结合循环语句实现

for i in range(65,91):

letter = chr(i)

print(letter,end=" ")

print()

3.计算出 100~999以内的水仙花数（水仙花是指百位的3次方加上十位的3次方加上个位的3次方等于原数字）

for x in range(100,1000):

s = str(x)

bai = int(s[0])

shi = int(s[1])

ge = int(s[2])

if s == str(bai \*\* 3 + shi \*\* 3 + ge\*\*3):

print("水仙花数",s)

May9

1 列表的 in / not in

1.判断一个数据元素是否存在于容器（列表）内，如果存在返回True，则返回False

2.not in 的返回值与in运算符相反

示例：

x = [1 , ‘Two’ , 3.14 , ‘四’]

1 in x #True

2 in x #False

3 not in x #True

‘四’ not in x #False

2 列表的 索引index / 切片 slice

列表的索引语句：

列表[整数表达式]

用法：

列表的索引取值与字符串的索引取值规则完全相同

列表的索引为分正向索引和反向索引

示例：

L = [‘A’ , 2 , ‘B’, 3]

print（L[1]） # 2

x = L[2] # x = ‘B’

3 列表的索引赋值

列表是可改变的序列，可以通过索引赋值改变列表中的元素

语法：

列表[索引] = 表达式

示例：

x = [1 , 2 , 3 , 4]

x[2] = 3.14 #改变了第三个元素的值

4 列表的切片

列表[:]

列表的[::]

列表的切片取值返回一个列表，规则统统与字符串的切片规则

示例：

x = list(range(9))

y = x[1:9:2] #y = [1,3,5,7]

列表的切片赋值语法：

列表[切片] = 可迭代对象

说明：

切片赋值的赋值运算符的右侧必须是一个可迭代对象

示例：

L = [2,3,4]

L[0:1] = [1.1, 2.2]

print(L) #[1.1, 2.2, 3 ,4]

L = [2 , 3 , 4]

L[:] = [7,8]

print(L) #[7,8]

>>> L=[2,3,4]

>>> L=[2,3,4]

>>> L[1:2]=[]

>>> L

[2, 4]

>>> L[1:1]=[2.1,2.2]

>>> L

[2, 2.1, 2.2, 3, 4]

4.1 切片步长不为1的切片赋值

L = list(range(1,9)) #[1,2,3,4,5,6,7,8,]

L [1::2] = [2.2 , 4.4 , 6.6 , 8.8]

print(L) # L= [1 , 2.2 , 3 , 4.4 , 5 , 6.6 , 7 , 8.8]

注意事项：

对于步长不等于1的切片赋值，赋值运算符的右侧的可迭代对象提供元素的个数一定要等于切片切出的段数，如：

L = [1,2,3,4,5,6,]

L[::2] = ‘ABCD’ #错误的输出

L[::2] = ‘ABC’ #正确 ['A', 2, 'B', 4, 'C', 6]

5 del语句用于删除列表元素

语法：

del 列表[索引]

del 列表[切片]

示例：

L = [1，2，3，4，5，6]

del L[0] #L = [2,3,4,5,6]

del L[-1] #L = [2,3,4,5]

L = [1,2,3,4,5,6]

del L[::2] #L = [2,4,6]

6 python3中常用的序列函数

|  |  |
| --- | --- |
| len(x) | 返回序列的长度 |
| max(x) | 返回序列的最大值元素 |
| min(x) | 返回序列中最小值元素 |
| sum(x) | 返回序列中所有元素的和（元素必须是数值类型） |
| any(x) | 真值测试，如果列表中任意一个值为真值，则返回True，否则返回False |
| all(x) | 真值测试，如果序列中所有值为真值，则返回True，只要有一个为假，则返回False |

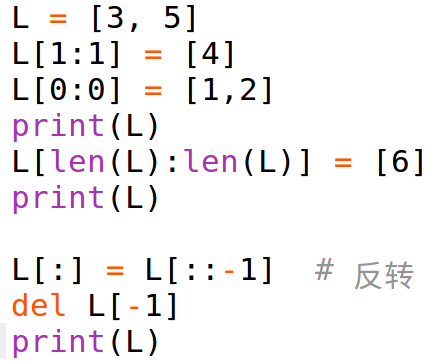
练习：

1.已知有列表

L = [3，5]

用索引和切片操作，将列表改为：L=[1，2，3，4，5，6]

将列表反转（前后对调），然后删除最后一个元素



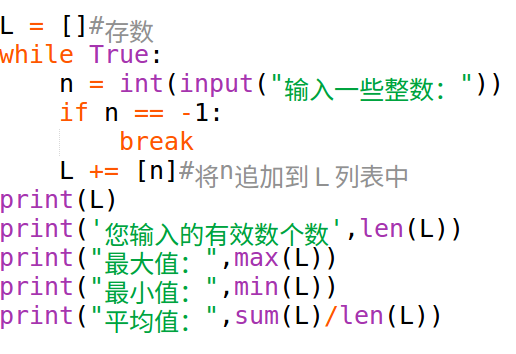
2.写程序，让用户循环输入一些整数，当输入-1时结束输入，将这些整数存于列表L中

1）打印您共输入了几个有效的数

2）打印您输入的数的最大值是多少？

3）打印您输入的数的最小数是多少？

4）打印您输入这些数的平均值



6 Python3中常用的列表方法（method)

* 见:help(list)

| **方法** | **意义** |
| --- | --- |
| L.index(v [, begin[, end]]) | 返回对应元素的索引下标, begin为开始索引，end为结  束索引,当 value 不存在时触发ValueError错误 |
| L.insert(index, obj) | 将某个元素插放到列表中指定的位置（index）obj（数） |
| L.count(x) | 返回列表中元素的个数 |
| L.remove(x) | 从列表中删除第一次出现在列表中的值x（x是数值不是索引） |
| L.copy() | 复制此列表（只复制一层，不会复制深层对象) |
| L.append(x) | 向列表中追加单个元素x（不是索引） |
| L.extend(lst) | 向列表追加另一个列表 |
| L.clear() | 清空列表,等同于 L[ : ] = [ ] |
| L.sort(reverse=False) | 将列表中的元素进行排序，默认顺序按值的小到大的顺序  排列 |
| L.reverse() | 列表的反转，用来改变原列表的先后顺序 |
| L.pop([index]) | 删除索引对应的元素，如果不加索引，默认删除最后元素，同时  返回删除元素的引用关系 |

练习：

写一个程序，输入多行文字，当输入空行时结束输入。将原输入的所有字符串存入列表L中

1）按原来输入行的顺序反向打印这些行

列：

输入：hello world

输入：welcome to China

输入：I like python

输入：<回车>

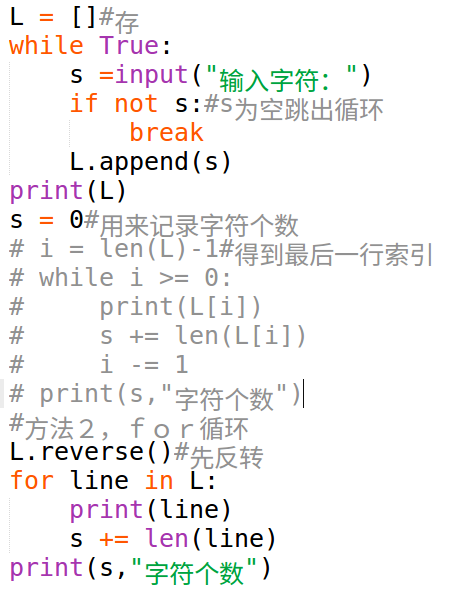
显示：

I like python

welcome to China

hello world

2）打印出您共输入了多少字符？



7 字符串文本解析方法split 和 join

S.split(sep=None)将字符串，使用sep作用为隔分隔符S字符串，返回分割后的字符串的列表，当不给定参数时，用空白字符作为分隔符进行分割。

S.join(iterable)用可迭代对象中的字符串，返回一个中间用S进行分隔的字符串

例：

s = ‘Beijing is capital’

L = s.split(‘’) #L = [‘Beijing’,’is’,’capital’]

s = ‘\\’

L = [‘C:’,’Programe files’,’python3’]

s2 = s.join(L) #s2 = b’C:\Programe files\python3’

8 深拷贝deep copy 和浅拷贝shallow copy

浅拷贝：

是指在复制过程中只复制一层变量，不会复制深层变量绑定的对象的复制过程。（跟之前的对象仍有连接，之前对象改变拷贝后的也改变）

L = [3.1,3.2]

L1 = [1,2,L]

L2 = L1.copy()

print(L1) #[1, 2, [3.1, 3.2]]

print(L2) #[1, 2, [3.1, 3.2]]

L2[2][0] = 3.14 #L2列表中第2个（列表[2]）替换列表中的第0[0]个

print(L1) #[1, 2, [3.14, 3.2]]

print(L2) #[1, 2, [3.14, 3.2]]

深拷贝：

import copy #导入copy模块

L = [3.1，3.2]

L1 = [1,2,L]

L2 = copy.deepcopy(L1) #深拷贝

print(L1) # [1, 2, [3.1, 3.2]]

print(L2) #[1, 2, [3.1, 3.2]]

L2[2][0] = 3.14

print(L1) #[1, 2, [3.1, 3.2]]

print(L2) #[1, 2, [3.14, 3.2]]

注：深拷贝通常只对可改变对象进行复制，不可改变对象通常不变

9 列表推导式list comprehensive

列表推导式是用可迭代对象依次生成带有多个元素的列表的表达式

作用：

用简易方法生成列表

语法：

[表达式 for 变量 in 可迭代对象]

或

[表达式 for 变量 in 可迭代对象 if 真值表达式]

示例：

#以下生成一个数值为1~19的平方列表

L = [x \* x for x in range(1,10)]

练习：

用列表推导式生成1~100内的奇数的列表[1,3,5,7,9,……]

L = [x for x in range(1,100,2)]

print(L)

L = [x for x in range(1,101) if x%2==1]

print(L)

9.1列表推导式的嵌套

语法：

[表达式1 for 变量1 in 可迭代对象1 if 真值表达式1

for 变量2 in 可迭代对象2 if 真值表达式2

……]

示例：

L1 = [2,3,5]

L2 = [7,11,13]

#将L1中的全部元素与L2中的全部元素依次相乘后放到列表L3中

L3 = [x \* y for x in L1 for y in L2]

print(L3)

练习：

1.e用字符串s = “ABC”和s2 = “123”生成如下列表：

[‘A1’,’A2’,’A3’,’B1’,’B2’,’B3’,’C1’,’C2’,’C3’]

2.有一些数字存在于列表L中，如：

L=[1,3,2,1,6,4,2，……98,82]此数据自己定义

将列表L中的数存入于另一个列表L2中（要求，重复出现多次的数只在L2列表中保留一份）

3.生成前40个斐波那契数（Fibonacci）1 1 2 3 5 8 13……

（自第三个起，之后的所有数为前两个数之和）要求，将这些数保存在列表中，最后打印列表中的这些数

提示：用循环、列表和变量组合可以实现

L = []

for i in range(41):

if i == 0 or i == 1:

L.append(i)

else:

L.append(L[i-2]+L[i-1])

print(L)

May10

1 元组tuple

元组是不可以改变的序列。同list一样，元组可以存放任意类型的元素。一旦元组生成，则它不可以改变。

元组的表示方式：

用小括号（）括起来，单个元素括起来用逗号（，）区分是单个对象还是元组

创建空元组的自符值:

t=()

创建非空元组的字面值:

t=200,

t=(20,)

t=(1,2,3)

t=100,200,300

元组的错误示例：

t = (20) #t是绑定整数的

x , y , z = 100 , 200 , 300 #序列复制

x , y , z = ‘ABC’

x , y , z = [10,20,30]

元组的构造函数tuple

tuple()生成一个空的元组，等同于()

tuple(icerable)用可迭代对象生成一个元组

示例：

t = tuple()

t = tuple(range(10))

t = tuple(‘hello’)

t = tuple([1,2,3,4])

元组的算数运算：

+ += \* \*=

用法与列表的用法完全相同

元组的比较运算：

< <= > >= == !=

in / not in

索引取值

切片取值

规则与列表完全相同

区别：

元组是不可改变的对象，不支持索引赋值和切片赋值

元组的方法：

T.count(value) -> integer -- return number of occurrences of value

T.index(value, [start, [stop]]) -> intege return first index of value.

可以用于序列的函数：

len ,max,min,sum,all ,any

----------------------以上用法和列表几乎相同----------------------

三个构造函数：

str（obj）

list(iterable)

tuple(iterable)

用于创建相应的对象

其他函数：

reversed(seq)返回反向顺序的可迭代对象

sorted(iterable,reversed=False)

返回已排序的列表

for x in reversed("ABCD"):

print(x) #DCBA

L = [8,6,3,5,7]

L2 = sorted(L)

print(L2)# [3, 5, 6, 7, 8]

2 字典dict

1.字典是一种可变的容器

2.字典中的每一个数据都是用’键’（key）进行索引，而不像序列可以用下标进行索引

3.字典的数据没有先后顺序关系，字典的存储是无序的

4.子典中的数据以键（key）-值（value）对进行映射存储

5.字典的’键’不能重复，且只能用不可变类型(可以用元组)作为字典的键

字典的字面值表示方式：

用{}括起来，以冒号（:）分割‘键——值‘ 对，各键值对用分号分隔开

创建空字典:d = { }

创建非空字典：d = {‘name’:’tarena’,’age’:15} len(d) # 2

d = {‘姓名’:’小张’} len(d) # 1

d = {1:’壹’,2:’贰’}

2.1字典的构造函数dict

dict()创建一个空字典，等同于{}

dict(iterable)用可迭代对象初始化一个字典

dict(\*\*kwargs)关键字传参形式生成一个字典

示例：（）

d = dict()

d = dict([(‘name’,’tarena’),(‘age’,15)])

d = dict(name = ‘tarena’,age = 15)

{'name': 'tarena', 'age': 15}

>>> d = dict(("AB","CD"))

>>> d

{'C': 'D', 'A': 'B'}

2.2 不可变类型

int,float,complex,bool,str,tuple,frozenset(固定集合),bytes(字节串)

2.3 可变类型

list,dict,set(集合),bytearray(字节数组)

2.4 字典的键索引

用[]运算符可以获取字典内的’键’所对应的’值’

语法：

字典[键]

获取数据元素

d = dict(name=’tarena’,age=15) # {'name': 'tarena', 'age': 15}

print(d[‘age’]) #15

2.5添加/修改字典元素

示例：

d = {}

d[‘name’] = ‘tarena’#创建一个新的键值对

d[‘age’] = 15 #创建键值对（如果已经存在，就会改变对应的值）

d[‘age’] = 16 #修改键值对

del语句删除字典的元素

语法：

del字典[键]

示例：

d = {‘name’:’China’,’pos’:’asia’}

del d[‘age’]

print(d)

del d[‘name’]

print(d)

2.6 字典中的 in/not in运算符

可以用in运算符来判断一个’键’是否存在于字典中，如果存在则返回True，否则返回False

not in 与 in返回值相反(字典的in/not in快于列表)

示例：

d = {‘a’:1,’b’:2}

‘a’ in d #True

1 in d #False

100 not in d #True

2 not in d #True

字典的迭代访问：

字典是可迭代对象，字典只能对键进行迭代访问

d ={‘name’:’tarena’,(2002,1,1):’生日’}

for x in d:

print(x)

name

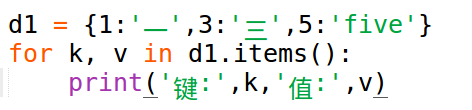
(2002, 1, 1)

2.7 可以用于字典的内建函数

|  |  |
| --- | --- |
| len（x） | 返回字典键-值对的数 |
| max（x） | 返回字典的键值的最大值 |
| min（x） | 返回字典的键的最小值 |
| sum（x） | 返回字典所有键的和 |
| any（x） | 真值测试，只对键测试，如果其中一个键为True，结果为True |
| all（x） | 真值测试，全部建为True时，结果才为True |

字典的方法：

| **函数** | **说明** |
| --- | --- |
| D代表字典对象 | |
| D.clear() | 清空字典 |
| D.pop(key) | 移除键，同时返回此键所对应的值 |
| D.copy() | 返回字典D的副本,只复制一层(浅拷贝) |
| D.update(D2) | 将字典 D2 合并到D中，如果键相同，则此键的值取D2的值作为新值 |
| D.get(key, default) | 返回键key所对应的值,如果没有此键，则返回default |
| D.keys() | 返回可迭代的 dict\_keys 集合对象 |
| D.values() | 返回可迭代的 dict\_values 值对象 |
| D.items() | 返回可迭代的 dict\_items 对象以元组的方式得到键-值对 |



键: 1 值: 一

键: 3 值: 三

键: 5 值: fived

练习：

1.写程序，实现下列需求：

将如下数据形成一个字典seasons

让用户输入一个整数代表这个季度，打印这个季度的信息，如果用户输入的信息不在字典内，则打印信息不存在。

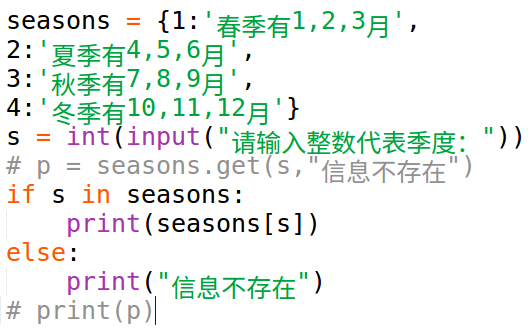
‘键’ ‘值’

1 ‘春季有1,2，3月’

2 ‘夏季有4,5,6月’

3 ‘秋季有7,8,9月’

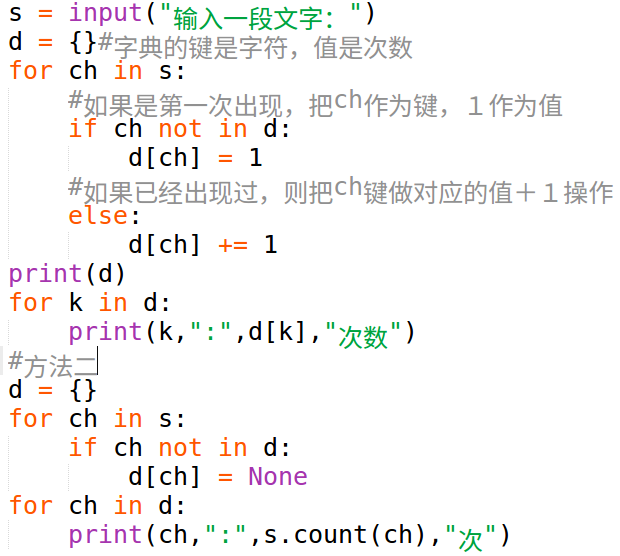
4 ‘冬季有10,11,12月’



2.写程序，输入地段字符串，打印出这个字符串中出现过的字符与出现次数：

如：输入：ABCDABCABA

输出：A:4次 C：2次 B：3次 D：1次



2.8 字典推导式

字典推导式是用可迭代对象依次生成字典内元素的表达式

语法：

{键表达式 : 真值表达式 for 变量 in 可迭代对象 [if 真值表达式]}

注：[]内的内容可以省略

示例：

d = {x : x\*\*2 for x in range(10)}

{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81}

d = {x : x\*\*2 for x in range(10) if x%3==0}

{0: 0, 9: 81, 3: 9, 6: 36}

练习：

1.有字符串列表如下：

L = [‘tarena’,’xiaozhang’,’xiaowang’]

生成如下字典:

d = {‘tarena’: 6 ,’xiaozhang’:9 , ‘xiaowang’:8}

注：字典的值是键的长度

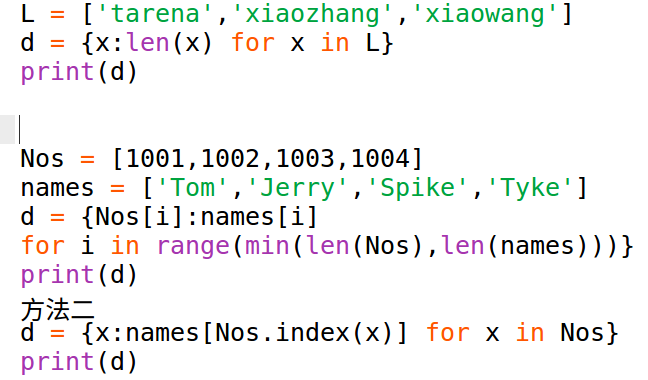
2.编号列表如下：

Nos = [1001,1002,1003,1004]

names=[‘Tom’,’Jerry’,’Spike’,’Tyke’]

生成了用Nos数据为键，以names为值的字典，如下：

{1001:’Tom’,1002:’Jerry’,……}



字典VS列表

|  |  |
| --- | --- |
| 相同点 | 不同点 |
| 都是可变的容器 | 索引方式不同（列表用整数索引，字典用键索引） |
|  | 字典的查找速度可能快于列表（重要） |
|  | 列表的存储是有序的，字典无序 |

练习：

1.写一个程序，输入一些单词和解释，将单词作为键，将解释作为值，将这些数据存入字典中。然后：输入查询的单词，显示出此单词的解释。

2.学生管理项目准备工作：

写一个程序，任意输入n个学生的信息，形成字典后存于列表中。

学生信息包括：姓名，年龄，成绩。循环输入学生信息，直到输入学生姓名为空时结束输入，最后形成字典如下：

L = [{‘name’:’xiaozhang’,’age’:20,’score’:100}

{‘name’:’xiaozhang’,’age’:21,’score’:89}

{‘name’:’xiaozhang’,’age’:19,’score’:98}]

将以上列表显示为如下表格：

+------------------+-------------+------------+

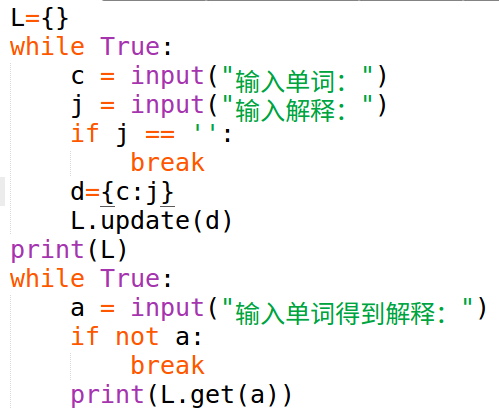
| name |age |score |

+------------------|--------------|------------+

|xiaozhang | 20 | 100 |

|xiaoli | | |

|xiaowang | | |  
+-------------------+---------------+-----------+



May11

1 集合set

1.1定义：

集合是可变的容器

集合内的数据对象是唯一的（不能重复多次的）

集合是无序的存储结构，集合中的数据没有先后顺序

集合内的元素必须是不可变的对象

集合是可迭代的（可以用for循环）

集合是相当于只有键没有值的字典（键就是集合的数据）

1.2集合的创建

创建空的集合

s = set（）

创建非空集合

s = {1,2,3}

2 集合的构造函数set

set（）创建空集合

set（iterable）用可迭代对象创建一个新的集合对象

示例：

s = set(“ABC”)

s = set(‘ABCCBA’)

s = set({1:”一”,2:”二”,5:”五”})

s = set([1,3.14,False])

s = set((2,3,5,7))

s = set([1,2,[3.1,3.2],4])错误的例子，因为列表[3.1,3.2]是可变的对象

3 集合的运算

交集，并集，补集，子集，超集

&生成两个集合的交集

s1 = {1,2,3}

s2 = {2,3,4}

s3 = s1 & s2 #{2, 3}

|生成两个集合的并集

s1 = {1,2,3}

s2 = {2,3,4}

s3 = s1|s2 #{1,2,3,4}

-生成两个集合的补集

s1 = {1,2,3}

s2 = {2,3,4}

s3 = s1 – s2 #{1}生成属于s1但不属于s2的所有元素的集合

^生成两个集合的对称补集

s1 = {1,2,3}

s2 = {2,3,4}

s3 = s1 ^ s2 #{1,4}等同于s3=(s1-s2)|(s2-s1)

<判断一个集合是另一个集合的子集

>判断一个集合是另一个集合的超集

s1 = {1,2,3}

s2 = {2,3}

s2 < s1 #True判断子集

s1 > s2 #True判断超集

== ！= #集合相同/不同

s1 = {1,2,3}

s2 = {2,3,1}

s1 == s2 #True

s1 != s2 #False

in /not in

等同于字典，in运算用于集合中，当某个值存在于集合中，则为真，否则为假

not in与in返回值相反

示例：

s = {1,’Two’,3.14}

1 in s #True

2 in s #False

3.14 not in s #False

4 not in s #True

4 python3中可以用于集合的函数

len(x),max(x),min(x),sun(x),any(x),all(x)

集合是可迭代对象

s = {1,2,3}

for x in s:

print(x) #1 2 3

练习：

经理：曹操，刘备，周瑜

技术员：曹操，周瑜，张飞，赵云

用集合1.即是经理也是技术员的有谁？

2.是经理，但不是技术员的有谁？

3.是技术员，不是经理的有谁？

4.张飞是经理吗？

5.身兼一职的人有谁

6.经理和技术员共有几个人？

s1 = {"曹操","刘备","周瑜"}

s2 = {"曹操","周瑜","张飞","赵云"}

print("即是经理也是技术员",s1&s2)

print("是经理但不是技术员",s1-s2)

print("是技术员，不是经理",s2-s1)

if "张飞" in s1:

print("张飞是经理")

else:

print("张飞不是经理")

print("身兼一职：",(s1|s2)-(s1&s2))

print("经理和技术员共有几人：",len(s1)+len(s2))

5 集合的方法

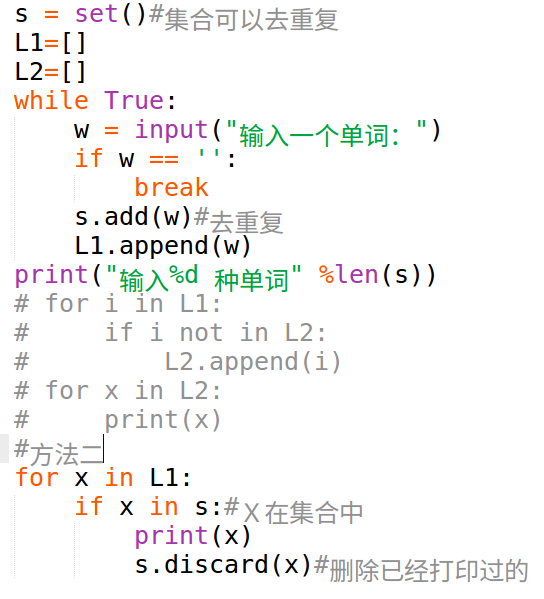
| **方法** | **意义** |
| --- | --- |
| S.add(e) | 在集合中添加一个新的元素e；如果元素已经存在，则不添加 |
| S.remove(e) | 从集合中删除一个元素，如果元素不存在于集合中，则会产  生一个KeyError错误 |
| S.discard(e) | 从集合S中移除一个元素e,在元素e不存在时什么都不做; |
| S.clear() | 清空集合内的所有元素 |
| S.copy() | 将集合进行一次浅拷贝 |
| S.pop() | 从集合S中删除一个随机元素;如果此集合为空，则引发  KeyError异常 |
| S.update(s2) | 用 S与s2得到的全集更新变量S |
| S.difference(s2) | 用S - s2 运算，返回存在于在S中，但不在s2中的所有元素  的集合 |
| S.difference\_update(s2) | 等同于 S = S - s2 |
| S.intersection(s2) | 等同于 S & s2 |
| S.intersection\_update(s2) | 等同于S = S & s2 |
| S.isdisjoint(s2) | 如果S与s2交集为空返回True,非空则返回False |
| S.issubset(s2) | 如果S与s2交集为非空返回True,空则返回False |
| S.issuperset(...) | 如果S为s2的子集返回True,否则返回False |
| S.symmetric\_difference(s2) | 返回对称补集,等同于 S ^ s2 |
| S.symmetric\_difference\_update(s2) | 用 S 与 s2 的对称补集更新 S |
| S.union(s2) | 生成 S 与 s2的全集 |

练习：任意输入一些单词，存入集合中，当输入空字符串时结束输入

1）打印您输入的单词的种类数（去重）

2）每个单词单词都打印到终端上显示

思考：如何让打印的次序和输入的次序一致



6 集合推导式

用可迭代对象创建（生成）集合的表达式

语法：

{表达式 for 变量 in 可迭代对象 [if 真值表达式]} []内可省去

示例：

>>> L = [2,3,5,7,7,3,5,7,11]

>>> s = {x\*\*2 for x in L}

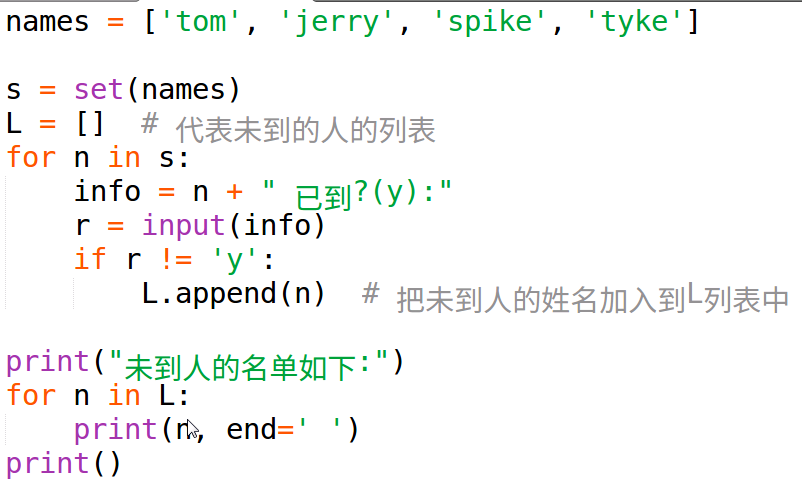
>>> s

{9, 4, 121, 25, 49}

集合推导式的嵌套规则与列表推导式相同

练习：

1.模拟一个点名系统，已知全班学生名单，随机打印学生的姓名进行点名，并得到此学生是否已到信息，输入‘y’代表已到，其他输出代表未到。点名结束后打印未到名单



问题：能用集合作为字典的键吗？

7 固定集合frozenset

固定集合是不可变，无序的，含有唯一元素的集合

作用：

固定集合可以作为字典的键，也可以作为集合的值（元素）

创建空的固定集合

fs = frozenset()

创建非空固定集合

fs = frozenset([2,3,5,7])

构造函数：

frozenset（）

frozenset（可迭代对象）#同set函数一致，返回固定集合

固定集合的运算：

&交集

|并集

-补集

^对称补集

> >= < <= == !=

in /not in

(以上运算规则等同于set中的用法)

固定集合的方法：

相当于集合的全部方法去掉修改集合的方法

阶段总结

1 数据类型

不可变类型：bool,int ,float,complex,str,tuple,frozenset,bytes

可变数据类型：list,dict,set,bytearray(字节数组)

值：None,True,Fasle……

2 运算符

+ - \* / // % \*\* > >= < <= == !=

is is not in not in not and or

& | ^

+(正号) -(负号)

3表达式

1

1+2

len([1,2,3])

条件表达式 x if x > y else y

全部的推导式：

列表，字典，集合推导式（三种）

表达式可以作为函数的参数，一定能返回一个值，一定在赋值语句右边

4语句

表达式语句：

print（“abc”） ‘hello’

赋值语句：（只是在创建变量和修改变量）

a = 100 a = b = c = 200 x,y = 100,200（序列赋值）

if语句

while语句

for语句（只能对可迭代对象循环）

break语句

continue语句（重新开始循环）

pass语句

del语句

5函数

内建函数：

len(x) max(x) min(x) sum(x) any(x) all(x)

bool(x) int(x) float(x) complex(x) str(x) list(x) tuple(x)

dict(x) set(x) frozenset(x)[横线上的为创建数据对象的函数]

abs(x)

round(x)四舍五入

pow(x,y……)求幂运算

bin(x)

oct(x)

hex(x)

chr(x)

ord(x)字符串转换为数字

range([start,]stop,[,step])

input(x)

print(……)