for … else 表示这样的意思，for 中的语句和普通的没有区别，else 中的语句会在循环正常执行完（即 for 不是通过 break 跳出而中断的）的情况下执行，while … else 也是一样

break语句，就像在C语言中，打破了最小封闭for或while循环

pass 一般用于占位置。因为如果定义一个空函数程序会报错，当你没有想好函数的内容是可以用 pass 填充，使程序可以正常运行.

{}是字典 dict.clear() del dict[‘a’] del dict

[]是数组

()是元组

使用"L"来显示长整型

dir(cmath) 详细信息

python中三引号可以将复杂的字符串进行复制:

python三引号允许一个字符串跨多行，字符串中可以包含换行符、制表符以及其他特殊字符

list1.append（） 使用 append() 添加元素

|  |  |  |
| --- | --- | --- |
| L[-2] | 'Runoob' | 读取列表中倒数第二个元素 |

Python的元组与列表类似，不同之处在于元组的元素不能修改。

元组使用小括号，列表使用方括号。

ticks=time.strftime("%Y",time.localtime())

try: 正常的操作 ......................

except(Exception1[, Exception2[,...ExceptionN]]]): 发生以上多个异常中的一个，执行这块代码 ......................

else: 如果没有异常执行这块代码

try-finally 语句无论是否发生异常都将执行最后的代码。

我们可以使用raise语句自己触发异常

**\_\_private\_method**：两个下划线开头，声明该方法为私有方法，不能在类的外部调用

re.match只匹配字符串的开始，如果字符串开始不符合正则表达式，则匹配失败，函数返回None；而re.search匹配整个字符串，直到找到一个匹配。

def priteme(\*str)://不定长参数。

如果要给函数内的全局变量赋值，必须使用 global 语句

Python 的 re 模块提供了re.sub用于替换字符串中的匹配项。

re.sub(pattern, repl, string, count=0, flags=0)

repl : 替换的字符串，也可为一个函数。

和 findall 类似，在字符串中找到正则表达式所匹配的所有子串，并把它们作为一个迭代器返回

for num in range(lower,upper+1):

print('{}x{}={} '.format(j, i, i\*j),end='')

print('{}{}{}\n'.format(i,j,k),end='')

if

elif

else

n=153 print(n/100) 1.53 print(n//100) 1

sum=(n//100)\*\*3+((n%100)//10)\*\*3+(n%10)\*\*3 153

print(bin(a))

print(hex(a))

print(oct(a))

print(ord(a)) 字符-》ASCII码

print(chr(a)) ASCII码-》字符

**print**(str.isalnum()) # 判断所有字符都是数字或者字母  
**print**(str.isalpha()) # 判断所有字符都是字母  
**print**(str.isdigit()) # 判断所有字符都是数字  
**print**(str.islower()) # 判断所有字符都是小写  
**print**(str.isupper()) # 判断所有字符都是大写  
**print**(str.istitle()) # 判断所有单词都是首字母大写，像标题  
**print**(str.isspace()) # 判断所有字符都是空白字符、\t、\n、\r

print(str.upper()) # 把所有字符中的小写字母转换成大写字母 print(str.lower()) # 把所有字符中的大写字母转换成小写字母 print(str.capitalize()) # 把第一个字母转化为大写字母，其余小写 print(str.title()) # 把每个单词的第一个字母转化为大写，其余小写

li=['a', 'b', 'mpilgrim', 'z', 'example']

li.append("new")添加一个

try: 正常逻辑

except Exception,err: 触发自定义异常

else: 其余代码

def mye( level ):

if level < 1:

raise Exception,"Invalid level!"

class Runoob:

\_\_site = "www.runoob.com"

runoob = Runoob()

print(runoob.\_Runoob\_\_site)

输出：[www.runoob.com](http://www.runoob.com)

class people():

def \_\_init\_\_(self,n,a,w):

self.name=n

self.age=a

self.weight=w

def speak(self):

print("%s speak 我 %d 岁"%(self.name,self.age))

class student(people):

def \_\_init\_\_(self,n,a,w,g):

self.grade=g

people.\_\_init\_\_(self,n,a,w)

def speak(self):

print("%s speak 我 %d 岁 在读 %d 年级"%(self.name,self.age,self.grade))

class speaker():

def \_\_init\_\_(self,n,t):

self.name=n

self.topic=t

def speak(self):

print("%s 说我今天演讲的主题是:%s"%(self.name,self.topic))

class sample(speaker,student):

\_\_hh='kkk'

def hh(self):

return self.\_\_hh

def \_\_init\_\_(self,n,a,w,g,t):

student.\_\_init\_\_(self,n,a,w,g)

speaker.\_\_init\_\_(self,n,t)

s=sample('bhh',33,123,4,'Python')

print(s.hh())

s.speak()

kkk

bhh 说我今天演讲的主题是:Python（就近的基类输出）

需要注意圆括号中父类的顺序，若是父类中有相同的方法名，而在子类使用时未指定，python从左至右搜索 即方法在子类中未找到时，从左到右查找父类中是否包含方法。

类定义了 \_\_init\_\_() 方法，类的实例化操作会自动调用 \_\_init\_\_() 方法。

**类的私有属性**

**\_\_private\_attrs**：两个下划线开头，声明该属性为私有，不能在类的外部被使用或直接访问。在类内部的方法中使用时 **self.\_\_private\_attrs**

**类的方法**

在类的内部，使用 def 关键字来定义一个方法，与一般函数定义不同，类方法必须包含参数 **self**，且为第一个参数

import os **操作系统接口**

import glob **文件通配符**

import sys **命令行参数**

import re **字符串正则匹配**

import math **数学**

import random andom提供了生成随机数的工具。

from urllib.request import urlopen **访问 互联网**

from datetime import date **日期和时间**

import zlib **数据压缩**

from timeit import Timer **性能度量**

import doctest **测试模块**

**Python 正则表达式{** re 模块**}**

re.match 尝试从字符串的起始位置匹配一个模式，如果不是起始位置匹配成功的话，match()就返回none。re.match(pattern, string, flags=0)

re.search 扫描整个字符串并返回第一个成功的匹配。

函数语法：re.search(pattern, string, flags=0)

import re

print(re.match('www', 'www.runoob.com').span()) # 在起始位置匹配

print(re.match('com', 'www.runoob.com')) # 不在起始位置匹配

line="Cats are smarter than dogs"

mathobj=re.match(r'(.\*) are (.\*?) ',line,re.M|re.I)// # re.I 表示忽略大小写

if(mathobj):

print(mathobj.group())

print(mathobj.group(1))

print(mathobj.group(2))

Python 的 re 模块提供了re.sub用于替换字符串中的匹配项。

# 将匹配的数字乘以 2

def double(matched):

value = int(matched.group('value'))

return str(value + 2)

s = 'A23G4HFD567'

print(re.sub('(?P<value>\d+)', double, s))

import re

pattern = re.compile(r'\d+') # 用于匹配至少一个数字

m = pattern.match('one12twothree34four') # 查找头部，没有匹配

print (m)

m = pattern.match('one12twothree34four', 3, 10)

print (m)

print(m.group(0))

print(m.start(0))

print(m.end(0))

print(3,5)

None

<\_sre.SRE\_Match object; span=(3, 5), match='12'>

12

3

5

3 5

match 和 search 是匹配一次 findall 匹配所有。

it = re.finditer(r"\d+","12a32bc43jf3")

for match in it:

print (match.group() )

12

32

43

3

100 例

1、for i in range(1,5):

for j in range(1,5):

for k in range(1,5):

if i!=k and j!=k and i!=j:

print('{}{}{}\n'.format(i,j,k),end='')

2、

profits=[1000000,600000,400000,200000,100000,0]

pushMoney=[0.01,0.015,0.03,0.05,0.075,0.1]

currentProfit=120000

sum=0

for index in range(0,6):

if currentProfit>profits[index]:

sum+=(currentProfit-profits[index])\*pushMoney[index]

print('提成：{} '.format(sum))

currentProfit=profits[index]

print('一共是：{} 元提成'.format(sum))

3、for i in range(1,100):

if(168%i==0):

j=168/i

if i>j and (i+j)%2==0 and (i-j)%2==0:

m=(i+j)/2

n=(i-j)/2

x=n\*n-100

print(x)

4、import time

print(time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time())))

time.sleep(5)

print(time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time())))

5、P1=1

P2=1

for i in range(1,3):

print('{}\t{}\t'.format(P1,P2),end="")

P1=P1+P2

P2=P1+P2

6、num=0

for i in range(101,201):

for j in range(2,i):

if(i%j==0):

break

else:

num+=1

print(i)

print('一共有 %d 个素数'%num)

6、

for i in range(100,9999):

t=i

nums=[]

while(t>0):

nums.append(t%10)

t//=10

uu=0

for k in nums:

uu+=k\*k\*k

if(uu==i):

print(i)

7、

input=int(90)

hh=int(input)

nums=[]

while(hh>2):

for i in range(2,100):

if(hh%i==0):

j=hh//i

nums.append(i)

print(i,j)

hh=int(j)

break

print(nums)

8、

s = '123runoobc kdf235\*(dfl'

letters = 0

space = 0

digit = 0

others = 0

for c in s:

if c.isalpha():

letters += 1

elif c.isspace():

space += 1

elif c.isdigit():

digit += 1

else:

others += 1

print('char = %d,space = %d,digit = %d,others = %d' % (letters,space,digit,others))

9、

import math

a=4

n=4

nums=0

for i in range(1,int(n+1)):

num=0

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

pows=a\*math.pow(10,j-1)

num+=int(pows)

print(num)

nums+=num

print(nums)

10、

for i in range(2,1000):

num=0

nums=[]

for j in range(1,i):

if(i%j==0):

nums.append(j)

num+=j

if(i==num):

print(i)

print(nums)

20

h=100

def meici(high):

return high/2

allh=0

for i in range(0,10):

if i==0:

allh+=h

else:

allh+=2\*h

h=meici(h)

print(h)

print(allh)

21\

sum=0

dangqian=1

for i in range(0,9):

qiyitian=(dangqian+1)\*2

dangqian=qiyitian

print(dangqian)

22

for i in range(ord('x'),ord('z')+1):

for j in range(ord('x'),ord('z')+1):

for k in range(ord('x'),ord('z')+1):

if i!=j and i!=k and j!=k and i!=ord('x') and k!=ord('x') and k!=ord('z'):

print(chr(i),chr(j),chr(k))

23

qiank=0

houk=0

xing=0

for i in range(1,5):

houk=qiank=4-i

xing=2\*(i-1)+1

for j in range(1,qiank+1):

print('{}'.format(' '),end='')

for j in range(1,xing+1):

print('{}'.format('\*'),end='')

print()

for i in range(1,4):

houk=qiank=i

xing=7-2\*i

for j in range(1,qiank+1):

print('{}'.format(' '),end='')

for j in range(1,xing+1):

print('{}'.format('\*'),end='')

print()

25

sum=0

num=1

for i in range(1,21):

num=i

for j in range(1,i):

num\*=j

sum+=num

print(sum)

26

def fact(x):

if x==1:

return x

else:

return x\*fact(x-1)

print(fact(5))

27

s='bianhuan'

l=len(s)

def output(s,l):

if l==0:

return

print(s[l-1])

output(s,l-1)

output(s,l)

27

num=0

def putout(x):

global num

if (x//10)==0:

num+=1

print(x)

else:

num+=1

remainder=x%10

print(remainder)

putout(x//10)

putout(123)

print(num)

30

num=0

nums=[]

def putout(x):

global num

if (x//10)==0:

num+=1

print(x)

nums.append(x)

else:

num+=1

remainder=x%10

print(remainder)

putout(x//10)

nums.append(remainder)

putout(12322)

print(num)

for i in range(0,2):

if nums[i]!=nums[4-i]:

print('No')

break

else:

print('Yes')

32

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

nums=lists[::-1]

print(lists)

print(nums)

33

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

sl='#'.join(str(n) for n in l)

print(sl)

34

def hello():

print("Hello")

def three\_hello():

for i in range(3):

hello()

if \_\_name\_\_=='\_\_main\_\_':

three\_hello()

37

nums=[2,5,9,4,44,67,12,88,19,70]

print(nums)

nums.sort()

print(nums)

38

a = [[2, 2, 2], [2, 2, 2], [2, 2, 2]]

a=[]

a.append([])

a[0].append(2)

a[0].append(3)

print(a) [[2, 3]]

a=[[78,34,23],[34,56,33],[12,21,2]]

sum=0

for i in range(3):

sum+=a[i][i]

print(sum)

39

a=[1,2,3,5,6,7,8]

a.append(4)

print(a)

a.sort()

print(a)

41

class static\_class:

var=5

def varfunc(self):

self.var+=1

print(self.var)

print(static\_class.var)

a=static\_class()

for i in range(3):

1. varfunc()

44

x=[[12,7,3],[4,5,6],[7,8,9]]

y=[[5,8,1],[6,7,3],[4,5,9]]

print(x)

print(y)

z=[]

for i in range(3):

z.append([])

for j in range(3):

num=x[i][j]+y[i][j]

z[i].append(num)

print(z)

46

num=10

def SQ(x):

return x\*x

again=1

while again:

if(SQ(num)>=50):

again=1

print("TRUE")

num-=1

else:

again=0

print("FALSE")

47

x=10

y=20

print('x=%d y=%d'%(x,y))

x,y=y,x

print('x=%d y=%d'%(x,y))

ambda表达式，通常是在**需要一个函数，但是又不想费神去命名一个函数**的场合下使用

add = lambda x, y : x+y add(1,2) # 结果为3

maxnum=lambda x,y: x if x>y else y

minnum=lambda x,y:(x>y)\*y+(x<y)\*x

print(maxnum(109,40))

print(minnum(109,40))

50

import random

x=random.uniform(10,20)

print(x)

51

a=9

b=2

c=a&b 按位与（c=a|b 按位或）（c=a^b 按位异或）（~0=1; ~1=0;按位取反）

print(c)

54

c = ~(~0 << 4)

print(c) 设置一个低4位全为1,其余全为0的数。可用~(~0<<4) 15

88

b=5

print(b\*'\*') 5个\*

93

import time start = time.clock()

for i in range(10000):

print i

end = time.clock()

print 'different is %6.3f' % (end - start)

96

a='shhdkhhkdihhdklhhk'

b='hh'

c=a.count(b)

print(c)

99

fp = open('test.txt','w')

fp = open('test.txt','r')

with open('test1.txt','w') as f:

f.write(str.upper())

a='jkkl'

b='TwEr'

l=list(a+b)

print(l)

l.sort()

s=''

s=s.join(l)

print(s)

s=s.upper()

print(s)

s=s.lower()

print(s)

100

i = ['a', 'b'] l = [1, 2] print dict([i,l])

排序。

def bubbleSort(arr):

for i in range(len(arr)):

for j in range(len(arr)-1):

if(arr[j]>arr[j+1]):

arr[j],arr[j+1]=arr[j+1],arr[j]

def selectSort(arr):

for i in range(len(arr)):

selectIndex=i

for j in range(i+1,len(arr)):

if arr[j]<arr[selectIndex]:

selectIndex=j

if selectIndex!=i:

arr[i],arr[selectIndex]=arr[selectIndex],arr[i]

def insertSort(arr):

for i in range(len(arr)):

preIndex=i-1

current=arr[i]

while preIndex>=0 and arr[preIndex]>current:

arr[preIndex+1]=arr[preIndex]

preIndex-=1

arr[preIndex+1]=current