## Python答辩题V9.8

1. Python你都会哪些？

我会（方法，程序控制【if…else ,for while】,面向对象，继承，封装，系统模块的使用[random,os ,smtplib,ddt,unittest……]，文件读取，数据库增删改查，excel表的读取，单元测试框架的封装）

1. 解释面向对象是什么意思？

面向对象主要是把事物对象化，包括其属性和行为

1. Python的优点和特点？
2. 语法简单：比如Python 不要求在每个语句的最后写分号，当然写上也没错；定义变量时不需要指明类型，甚至可以给同一个变量赋值不同类型的数据。
3. ① 程序员使用 Python 编写的代码是开源的。② Python 解释器和模块是开源的。
4. Python 是免费的
5. Python 是高级语言
6. Python 是解释型语言，能跨平台
7. Python 是面向对象的编程语言
8. Python 功能强大（模块众多）
9. Python 可扩展性强
10. 基础,python的数据类型？

7种，不可变数据类型。

数字，字符串，列表，元组，字典，集合，布尔

不可变数据：数字，字符串，元组

1、int -->将字符串数据类型转为int类型, 注：字符串内的内容必须是数字

2、对于布尔值，只有两种结果即True和False，其分别对应与二进制中的0和1。而对于真即True的值太多了，

我们只需要了解假即Flase的值有哪些---》None、空（即 [ ]/( ) /" "/{ }）、0；

3、str类：关于字符串是Python中最常用的数据类型，其用途也很多，我们可以使用单引号 ‘’或者双引号“”来创建字符串。

字符串是不可修改的。所有关于字符我们可以从 索引、切片、长度、遍历、删除、分割、清除空白、大小写转换、判断以什么开头等方面对字符串进行介绍。

4、列表list类：列表是由一系列特定元素顺序排列的元素组成的，它的元素可以是任何数据类型即数字、字符串、列表、元组、字典、布尔值等等，同时其元素也是可修改的。

5、元组tuple类：元组即为不可修改的列表。其于特性跟list相似。其使用圆括号而不是方括号来标识。

6、字典dict类:字典为一系列的键-值对，每个键值对用逗号隔开，每个键都与一个值相对应，可以通过使用键来访问对应的值。无序的。

键的定义必须是不可变的，即可以是数字、字符串也可以是元组，还有布尔值等。而值的定义可以是任意数据类型。

7、集合set类：关于集合set的定义：在我看来集合就像一个篮子，你可以往里面存东西也可往里面取东西，但是这些东西又是无序的，你很难指定单独去取某一样东西；同时它又可以通过一定的方法筛选去获得你需要的那部分东西。故集合可以 创建、增、删、关系运算。

集合的特性：

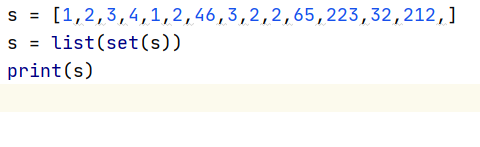
1、去重

2、无序

3、每个元素必须为不可变类型即（hashable类型，可作为字典的key）

1. 能不能将列表list = [1,3,4,5,2,6,1]快速去重？

Set (list)



1. (1)是不是元组？如果不是怎么才能变成元组呢？

（1,）

1. 列表中的深拷贝和浅拷贝的区别？

import copy  
li = [1,2,3,[5,6,7]]  
li1 = copy.deepcopy(li)*# 浅拷贝 [1,2,3,[4,6,7]]*li1[3][0] = 4  
  
print(li[3][0])

浅拷贝：无法拷贝列表中的子列表

深拷贝：完全拷贝

1. 列表与元组的区别？

列表是动态数组，它可变且可以重设长度

元组是静态数组，它不可变，其内部数据一旦创建便无法改变

列表：

1.可以增加列表内容 append

2.可以统计某个列表段在整个列表中出现的次数 count

3.可以插入一个字符串，并把整个字符串的每个字母拆分当作一个列表段追加到列表当中 extedn

4.可以查询某个列表段在整个列表的位置 index

5.可以在指定位置插入一个列表段 insert

6.可以删除列表的最后一个列表段 pop

7.可以删除指定列表中的某个列表段 remove

8.可以正向反向排序 reverse

9.可以按字母或数字排序 sort

10.定义列表时候使用中括号"[]"

注意：在列表当中，假如某两个列表段相同，不管是使用index还是remove都是统计的最靠前的列表段

元组：

1.可以统计某个元组段在整个元组中出现的次数 count

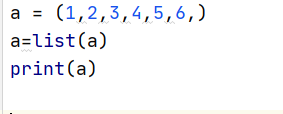
2.可以查询某个元组段在整个元组中的元组号 index

3.定义元组时候使用小括号

1. 能不能将元组快速转换成列表？

A = (1,4,5)

List(A)



1. 写个算法呗？冒泡排序，选择排序，列表翻转，字符串字符的统计，每隔60秒读取日志文件数据。

冒泡排序：a = [1,2,4,5,6,3,3,2,6,2,2]  
for i in range(len(a)-1):  
 for j in range(len(a)-1-i):  
 if a[j]>a[j+1]:  
 a[j],a[j+1]=a[j+1],a[j]  
print(a)

选择排序：number = [3,5,2,6,1,12,4,6,2]  
for i in range(len(number)):  
 for j in range(i+1,len(number)):  
 if number[i]>number[j]:  
 number[i],number[j]=number[j],number[i]  
print(number)

列表反转：

list=[2,1,3,5,8,8,8,8,5,9]  
list2=[]  
for i in range(len(list)):  
 list2.append(list.pop())  
print(list2)

字符串统计：string=('this is a dog,that is a monky!')

li=list(string)        # 将字符转换成列表

for i in range(0,len(li)):

    count=0

    flag=True

    for k in range(0,i):

        if li[k]==li[i]:

            flag=False

            break

    if flag==False:

        continue

    for j in range(0,len(li)):

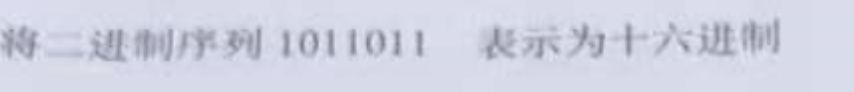
        if li[i]==li[j]:

            count += 1

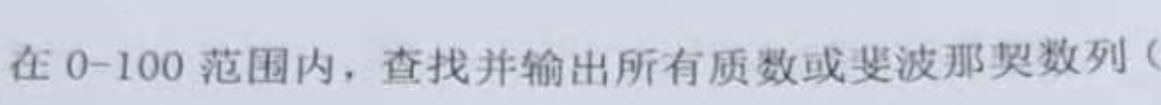
print(li[i],'出现了',count,'次！')

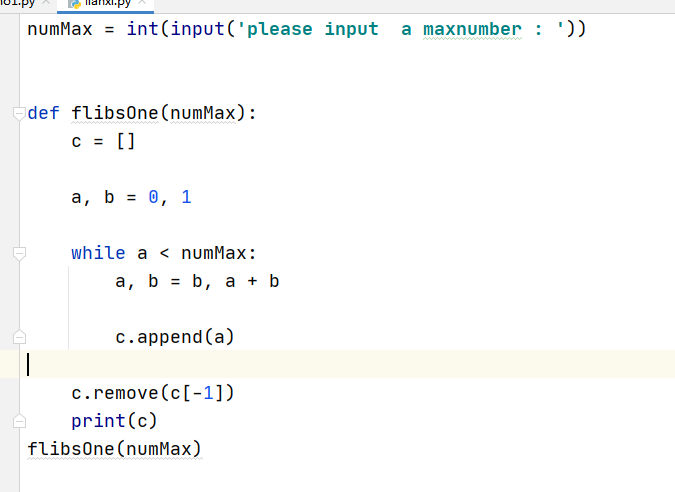
str = **'Great automation'**for index, ch in enumerate(str):  
 if ch in str[:index]:  
 continue  
 print(ch,**'有几次'**,str.count(ch))

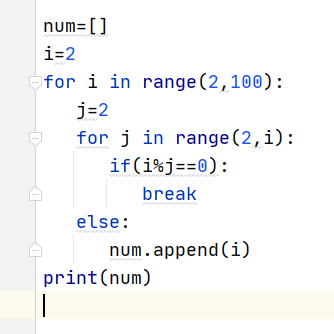
60秒读取  
import time  
class Wenjian:  
 def show(self):  
 while True:  
 f = open(**"baidu\_x\_system.log"**, **"r+"**, encoding=**"utf-8"**)  
 print(**"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**)  
 print(str(f.read()))  
 time.sleep(60)  
 f.close()  
w=Wenjian()  
w.show()

1. 

5b  
def he(str0):  
   strw =str(hex(int(str0 ,2))).replace('0x' ,'').upper()  
   return strw  
str1 ='1011011'  
print('%s二进制转十六进制为：'%str1 ,he(str1))

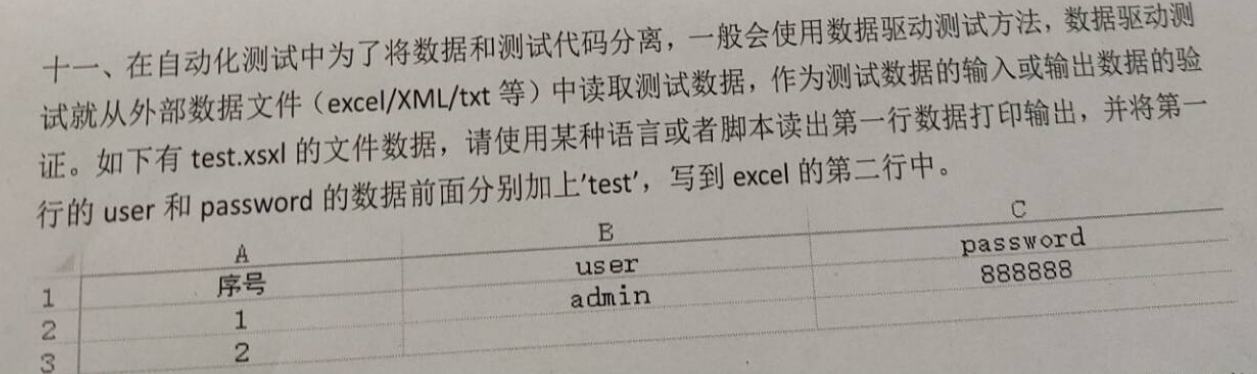
1. 

质数：

斐波那契数列：

1. 如何用面向对象描述一个水杯？

它是个容器，白色的，功能是容纳液体，最多可以容纳150ML的液体

1. 

import xlwt  
import xlrd  
workbook = xlwt.Workbook(encoding = **'utf-8'**)  
worksheet = workbook.add\_sheet(**'sheet1'**)  
worksheet.write(0,0, label = **"序号"**)  
worksheet.write(0,1, label = **"user"**)  
worksheet.write(0,2, label = **"password"**)  
worksheet.write(1,0, label = **"1"**)  
worksheet.write(1,1, label = **"admin"**)  
worksheet.write(1,2, label = **"888888"**)  
worksheet.write(2,0, label = **'2'**)  
worksheet.write(2,1, label = **'testadmin'**)  
worksheet.write(2,2, label = **'test888888'**)  
workbook.save(**'test.xls'**)  
data=xlrd.open\_workbook(**"test.xls"**)  
sheet=data.sheet\_by\_name(**"sheet1"**)  
print(sheet.row\_values(1))

1. Python的lambda表达式？lambda表达式写一个遍历列表？

lambda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，也就是指匿名函数。

lambda所表示的匿名函数的内容应该是很简单的，如果复杂的话，干脆就重新定义一个函数了，使用lambda就有点过于执拗了。

lambda就是用来定义一个匿名函数的，如果还要给他绑定一个名字的话，就会显得有点画蛇添足，通常是直接使用lambda函数。如下所示：

add = lambda x, y : x+y

add(1,2) # 结果为3

1. 什么是继承？继承有几种方式？继承的好处和坏处

继承是子类复用父类的属性和机制

两种

继承的好处：1. 提高了代码的复用性，维护性，让类于类之间产生了关系，是多态的前提

继承的弊端：类的耦合性增强了，开发的原则：高内聚，低耦合

1. Python中如何实现多线程？多线程中的start和run方法的区别？

通过threading.Thread进行创建多线程

Stat方法是启动一个子线程，在该子线程中执行run方法

Run方法并不启动一个新线程，就是在主线程中调用了一个普通函数而已

1. Python是如何管理内存？

从三个方面来讲一对象的引用计数机制，二垃圾回收机制，三内存池机制

1. 对象的引用计数机制

Python内部使用引用计数，来保持追踪内存中的对象，所有对象都有引用计数

引用计数增加的情况：

1. 一个对象分配一个新名称
2. 将其放入一个容器内

引用计数减少的情况；

1. 使用del语句对对象别名显示销毁
2. 引用超出作用域或被重新赋值
3. 垃圾回收

1．当一个对象的引用计数归零时，它将被垃圾收集机制处理掉。

2．当两个对象a和b相互引用时，del语句可以减少a和b的引用计数，并 销毁用于引用底层对象的名称。然而由于每个对象都包含一个对其他对象的应用，因此引用计数不会归零，对象也不会销毁。（从而导致内存泄露）。为解决这一问题，解释器会定期执行一个循环检测器，搜索不可访问对象的循环并删除它们。

三．内存机池机制

Python提供了对内存的垃圾收集机制，但是它将不用的内存放到内存池而不是返回给操作系统。

1，Pymalloc机制。为了加速Python的执行效率，Python引入了一个内存池机制，用于管理对小块内存的申请和释放。

2，Python中所有小于256个字节的对象都使用pymalloc实现的分配器，而大的对象则使用系统的malloc。

3，对于Python对象，如整数，浮点数和List，都有其独立的私有内存池，对象间不共享他们的内存池。也就是说如果你分配又释放了大量的整数，用于缓存这些整数的内存就不能再分配给浮点数。

1. 方法中\*args与\*\*kwargs的区别？

在传入多个实参时，\*args会把多出来的位置参数转化为tuple

\*\*kwargs会把关键字的参数转化为dict

1. 什么是python的字典？字典的键能用列表做键吗?

字典就是许多值的集合，字典的下标称为‘键’它所对应的值称为‘值’两个合起来统称为‘键—值‘

Python不允许用可变数据类型做键。

1. 如何删除字符串两边的空格？ “ hello world ” ,能讲讲str里的常用的api方法么？如何将字符串”3”转换成数字3?

“3” 🡪3 int()

3 🡪”3” str()

1. pass关键字的作用？如何申明一个变量为全局变量？

Pass 是空语句占位，是为了保持程序结构的完整性

申明全局变量使用关键字global

1. 有几种进制？

|  |  |  |  |
| --- | --- | --- | --- |
| 二进制 | 八进制 | 十进制 | 十六进制 |
| 0b | 0o |  | 0x |
| 0b01 | 0o76420 | 672864892789 | 0xf4e3 |

1. 什么是迭代器，和生成器？

迭代：顾名思义就是重复做一些事很多次（就现在循环中做的那样）。迭代器是实现了\_\_next\_\_()方法的对象（这个方法在调用时不需要任何参数），它是访问可迭代序列的一种方式，

通常其从序列的第一个元素开始访问，直到所有的元素都被访问才结束。

[注意]：迭代器只能前进不能后退

[迭代器的优点]:

使用迭代器不要求事先准备好整个迭代过程中的所有元素。迭代器仅仅在迭代到某个元素时才计算该元素，而在这之前或之后元素可以不存在或者被销毁。因此迭代器适合遍历一些数量巨大甚至无限的序列。

迭代器即迭代的工具

生成器也是迭代器，含有yield关键字的函数就是生成器函数

1. 文件读写？（步骤）有几种操作模式？（r,w,+,a,b）。用过哪些API？你用在测试的什么地方？

1. 打开文件 open（path，flag，encoding）

Path：打开路径

Flag ：打开方式（r，w，+，a，b）

Encoding ：编码方式

2 读取文件

3 关闭文件

Open（），read（），readline（），readlines（），close（），write（）

1. 所有类的父类？

Object

1. Python2.x 与python3.x区别？

打开文件不在支持file方法，只能用open

Range不在返回列表，而是一个可迭代的range对象

除法/不在是整除，而是的得到浮点数，整除需要用//

字符串及编码有相关的大变动，简单来说就是原来的str变成新的bytes，原来的unicode变成新的str

1. Python如何处理异常？

Raise

Try.

.except.finally

1. PDBC技术的操作步骤？获取所有数据?
2. 联网安装pymysql
3. 导入pymysql
4. 获取数据连接
5. 获取游标
6. 编写sql
7. 执行sql
8. 处理结果
9. 关闭资源

书写sql（select）语句，通过游标执行sql语句提取游标内的查询数据fetchall提取所有数据

1. 参数化怎么做(步骤)？用过哪些注解？
2. 首先导入ddt 的 ddt data unpack注解包
3. 将测试类用@ddt进行注释
4. 在要进行参数化的方法上：@ddt（\*数据源），unpack

@ddt @data @unpack

1. 如何使用测试集来测试？
2. 创建测试集
3. 用测试集加载测试用例
4. 创建HTML运行器
5. 用运行器运行测试集
6. 怎么生成测试报告？

import unittest  
import os  
import smtplib  
from email.mime.text import MIMEText  
from email.mime.multipart import MIMEMultipart  
from email.header import Header  
from HTMLTestRunner import HTMLTestRunner  
suite = unittest.TestSuite()  
  
loader = unittest.defaultTestLoader  
  
cases = loader.discover(os.getcwd()+**"**\\**testcase"**,pattern=**"\*.py"**)  
suite.addTest(cases)  
with open(**"银行开户测试报告.html"**,**"w+"**,encoding=**"utf-8"**) as f:  
 runner = HTMLTestRunner.HTMLTestRunner(  
 stream=f,  
 verbosity=1,  
 title=**"银行开户测试测试报告"**,  
 description=**"这是第三次迭代测试"** )  
 runner.run(suite)

1. 断言？多列举几个

assertEquals() assertNone()

1. 你的单元测试框架分为哪些模块？写个大致代码？数据读取工具写？邮件发送模块？

1.获取数据库连接(host地址，user用户，password密码，database数据库，charset编码集)

2.获取游标(所有sql语句等数据库的操作必须经过游标)

3.书写sql 语句

4.使用游标执行sql语句

    4.1 若做的是增删改，没有返回结果

    4.2 若做的是查询操作，有返回结果

5.处理执行后的结果

6.关闭游标

7.关闭连接

# 查询数据

# 1. 获取数据库 连接

con = pymysql.connect(host="localhost",user="root",password="",database="lttest",charset="utf8")

# 2.通过连接来获取游标

cursor = con.cursor()

# 3.书写sql

sql = "select \* from  person"

# 4.通过游标执行sql语句

s = cursor.execute(sql)

print(s)

# 提取游标内的查询数据

one = cursor.fetchmany(6)

print(list(one))

# 4.1 提交刚才执行操作，提交给数据

con.commit()

# 5.关闭资源

cursor.close()

con.close()