1，分享下看过的源代码或做项目过程中遇到的问题，怎么解决的?

答，项目遇到的问题用户权限验证，用装饰器。

2，多进程中如果一个进程死掉，怎么发现和处理。

答，用log日志模块，监控，或在每个进程下面判断用print打印。

3，用shell查询文件中文件并排序

find / -name 'file'

sort file

4，写一个生成器

# 简单的生成器函数

def rev\_str(my\_str):

length=len(my\_str)

for i in range(length-1,-1,-1):

yield my\_str[i]

for char in rev\_str("hello"):

print(char)

5，MySQL mongodb redis 数据库的端口都是什么

MySQL:3306 mongodb:28017 redis:6379

6，说几个linux命令

pwd ls kill mkdir

7，git 和svn 之间的区别

(1).GIT是分布式的，SVN不是：

这是GIT和其它非分布式的版本控制系统，例如SVN，CVS等，最核心的区别。如果你能理解这个概念，那么你就已经上手一半了。需要做一点声明，GIT并不是目前第一个或唯

一的分布式版本控制系统。还有一些系统，例如Bitkeeper, Mercurial等，也是运行在分布式模式上的。但GIT在这方面做的更好，而且有更多强大的功能特征。

GIT跟SVN一样有自己的集中式版本库或服务器。但，GIT更倾向于被使用于分布式模式，也就是每个开发人员从中心版本库/服务器上chect out代码后会在自己的机器上克隆

一个自己的版本库。可以这样说，如果你被困在一个不能连接网络的地方时，就像在飞机上，地下室，电梯里等，你仍然能够提交文件，查看历史版本记录，创建项目分支，

等。对一些人来说，这好像没多大用处，但当你突然遇到没有网络的环境时，这个将解决你的大麻烦。

同样，这种分布式的操作模式对于开源软件社区的开发来说也是个巨大的恩赐，你不必再像以前那样做出补丁包，通过email方式发送出去，你只需要创建一个分支，向项目

团队发送一个推请求。这能让你的代码保持最新，而且不会在传输过程中丢失。GitHub.com就是一个这样的优秀案例。

(2).GIT把内容按元数据方式存储，而SVN是按文件：

所有的资源控制系统都是把文件的元信息隐藏在一个类似.svn,.cvs等的文件夹里。如果你把.git目录的体积大小跟.svn比较，你会发现它们差距很大。因为,.git目录是处于

你的机器上的一个克隆版的版本库，它拥有中心版本库上所有的东西，例如标签，分支，版本记录等。

(3).GIT分支和SVN的分支不同：

分支在SVN中一点不特别，就是版本库中的另外的一个目录。如果你想知道是否合并了一个分支，你需要手工运行像这样的命令svn propget svn:mergeinfo，来确认代码是否

被合并。感谢Ben同学指出这个特征。所以，经常会发生有些分支被遗漏的情况。

然而，处理GIT的分支却是相当的简单和有趣。你可以从同一个工作目录下快速的在几个分支间切换。你很容易发现未被合并的分支，你能简单而快捷的合并这些文件。

(4).GIT没有一个全局的版本号，而SVN有：

目前为止这是跟SVN相比GIT缺少的最大的一个特征。你也知道，SVN的版本号实际是任何一个相应时间的源代码快照。我认为它是从CVS进化到SVN的最大的一个突破。因为GIT

和SVN从概念上就不同，我不知道GIT里是什么特征与之对应。如果你有任何的线索，请在评论里奉献出来与大家共享。

更新：有些读者指出，我们可以使用GIT的SHA-1来唯一的标识一个代码快照。这个并不能完全的代替SVN里容易阅读的数字版本号。但，用途应该是相同的。

(5).GIT的内容完整性要优于SVN：

GIT的内容存储使用的是SHA-1哈希算法。这能确保代码内容的完整性，确保在遇到磁盘故障和网络问题时降低对版本库的破坏。这里有一个很好的关于GIT内容完整性的讨论

–http://stackoverflow.com/questions/964331/git-file-integrity

8，说一说写项目中遇到的困难,公司中遇到的不好解决的问题

1 .Session 和 cookie 的作用？

答：session 主要是在服务器端存储用户的信息，用于对用户身份的验证，cookie主要客户端存储用户的基本信息，当客户端与服务器端再次进行数据交换时，服务器端的

session 可以使客户端与服务器端不用重新建立连接。

2 .怎么使float型数据四舍五入取整

答：round(x)

3 .怎么查看文件的大小

答：Ls -l

4 .怎么对list进行排序

答：l.sort()

5 .Django采用是什么框架模式

答：django是一种重量级的python web框架，采用的mtv框架模式

面试问题

1. django项目怎么实现的，遇到的问题是什么

2.tcp,udp

3.三次握手和四次挥手

三次握手

客户端通过向服务器端发送一个 SYN 来创建一个主动打开，作为三路握手的一部

分。客户端把这段连接的序号设定为随机数?A。

服务器端应当为一个合法的 SYN 回送一个 SYN/ACK。ACK?的确认码应为?A+1，

SYN/ACK?包本身又有一个随机序号?B。

最后，客户端再发送一个 ACK。当服务端受到这个 ACK 的时候，就完成了三路握

手，并进入了连接创建状态。此时包序号被设定为收到的确认号?A+1，而响应则

为 B+1。

四次挥手

由于 TCP 连接是全双工的，因此每个方向都必须单独进行关闭。这原则是当一方

完成它的数据发送任务后就能发送一个 FIN 来终止这个方向的连接。收到一个

FIN 只意味着这一方向上没有数据流动，一个 TCP 连接在收到一个 FIN 后仍能发

送数据。首先进行关闭的一方将执行主动关闭，而另一方执行被动关闭。

a.TCP 客户端发送一个 FIN，用来关闭客户到服务器的数据传送。

b.服务器收到这个 FIN，它发回一个 ACK，确认序号为收到的序号加 1。和 SYN一样，一个 FIN 将占用一个序号。

c.服务器关闭客户端的连接，发送一个 FIN 给客户端。

d.客户端发回 ACK 报文确认，并将确认序号设置为收到序号加 1。

4.请使用linux命令去编写爬虫

(1).linux下安装virtualenv sudo apt-get install python-virtualenv pip install virtualenvwrapper

(2).创建 py2和py3虚拟环境 virtualenv -p /user/bin/python3 py3

(3).windows pip install virtualenv

(4).virtualenv scrapytest 在当前创建虚拟环境 进入虚拟环境 cd Scripts 运行active.bat 推出deactive.bat

(5).新建py3虚拟环境 virtualenv -p py3路径/python.exe py3

1.虚拟环境的安装。

1.安装 pip install virtualenv

1.他的命令 virtualenv py2 ->在当前目录下创建一个虚拟环境

2.virtualenv -p py2-路径/python.exe py2 创建一个py2的虚拟环境

2.pip intall virtualenvwrapper

1.workon 列出所有虚拟环境 workon name 执行虚拟环境

2.mkvirtualenv name 默认路径：C:Users/Administrator/Envs 自己设置 PATH-> WORKON\_HOME == e:/

3.workon 不起作用时可以将py27下的Scrapys下的 workon.bat mkvirtual.bat 导入

4.mkvirtualenv --python=C:\Users\liuyc\AppData\Local\Programs\Python\Python35-32\python.exe mysearch

mkvirtualenv --python=C:\Python27\python.exe flask\_test

2.安装 scrapy

1.pip install -i https://pypi.douban.com/simple scrapy

2.www.lfd.uci.edu/~gohlke/pythonlibs/ 在这个网址下载Twisted 35-32版本 pip install ..安装

3.创建一个scrapy scrapy startproject ArticleSpider

4.模板创建 scrapy genspider jobbole blog.jobbole.com

5.pip install -i https://pypi.douban.com/simple/ pypiwin32

3.调试命令

1.scrapy shell http://python.jobbole.com/87566/

4.爬取拉钩网

1.scrapy genspider --list

2.scrapy genspider -t crawl lagou www.lagou.com

5.二叉树，排序

class mybtree: #创建一个类

class node:

def \_\_init\_\_(self):

self.data = None #给一个接收的值

self.left = None #左边的值

self.right = None #右边的值

def add(self,n):

if self.data > n.data: #self.data代表第一个值，n.data代表第二个值,当第一个值大于第二个值，走if

if self.left is None: #第一个值的左边为空时，走if

self.left = n

else:

self.left.add(n)

if self.data < n.data: #self.data代表第一个值，n.data代表第二个值,当第一个值小于第二个值，走if

if self.right is None: #第一个值的右边为空时，走if

self.right = n

else:

self.right.add(n)

def zhong(self): #中序

if self.left is not None: #左边的值为空时，走if

self.left.zhong()

print(self.data) #输出

if self.right is not None: #右边的值为空时，走if

self.right.zhong()

def \_\_init\_\_(self):

self.root = None

def add(self,data):

n = self.node()

n.data = data

if self.root is None:

self.root = n

else:

self.root.add(n)

def zhong(self):

self.root.zhong()

t = mybtree() #调用方法

t.add(11)

t.add(7)

t.add(18)

t.add(5)

t.add(8)

t.add(24)

t.add(2)

t.add(3)

t.zhong()

输出结果是：

2

3

5

7

8

11

18

24

用了几年linux

1.说几个Linux的高级命令

(1)cut 命令：提取文件中指定的字段(列)、字符内容。单词的默认分隔符为空格。

cut [选项]... [文件]...

选项：

-b 按照字节切

-c 按照字符切

-d 指定分割符

-f 指定以分割符分割的哪个字段

(2)sort 排序(升序、降序)

-n, --numeric-sort 根据数值比较，默认是按单个字符来比较

-r, --reverse 逆序输出排序结果

-t 指定分割符

-k 排序键(即排序的列字段)

-u 去掉重复行

(3)wc 统计行数 单词数 字节数

wc /etc/passwd

(4)uniq 合并连续重复的行

-u 只显示没有连续重复的行

-c 统计连续重复行的次数

-d 只显示连续重复行一次，哪一行连续重复了就显示哪行

-i 忽略大小写

2.说几个GIT的高级命令

(1)使用rebase而非merge来拉取上游修改

$ git checkout feature

$ git rebase master

(2) 克隆一个特定的远程分支

$ git init

$ git remote add -t -f origin

$ git checkout

(3)将cherry-pick远程提交合并到自己的分支中

$ git cherry-pick

(4)应用来自于不相关的本地仓库的补丁

$ git --git-dir=/.git format-patch -k -1 --stdout | git am -3 -k

3.简单说一下IO多路复用

多路复用IO也是阻塞IO，只是阻塞的方法是select/poll/epoll。select/epoll的好处就在于单个process就可以同时处理多个网络连接的IO。它的基本原理是select/epoll这

个函数会不断轮询所负责的IO操作，当某个IO操作有数据到达时，就通知用户进程。然后由用户进程去操作IO

4.对Flask的了解

Flask 是一种具有平缓学习曲线和庞大社区支持的微框架，利用它可以构建大规模的web应用。是搭建社区平台的神器之一。

利用它可以构建大规模的web应用。学习上手Flask非常轻松，但要深入理解却并不容易。本书从一个简单的Flask应用开始，通过解决若干实战中的问题，对一系列进阶的话

题进行了探讨。书中使用MVC（模型-视图-控制器）架构对示例应用进行了转化重构，以演示如何正确地组织应用代码结构。有了可扩展性强的应用结构之后，接下来的章节

使用Flask扩展为应用提供了额外的功能，包括用户登录和注册、NoSQL查询、REST API、一套后台管理界面，以及其他特性。然后，你会学到如何使用单元测试，保障代码持

续按照正确的方式工作，避免极具风险的猜测式编程。

一个简单的Flask 项目入手，由浅入深地探讨了一系列实战问题，包括如何使用SQLAlchemy 和Jinja 等工具进行Web 开发；如何正确地设计扩展性强的Flask 应用架构和搭

建MVC 环境；对于各种NoSQL 数据库的特性，何时应该、何时不应该及如何使用它们；通过使用Flask 扩展快速实现用户的身份系统、RESTful API、NoSQL查询、后台管理等

功能；如何创建自己的扩展；使用Celery 编写异步任务，使用pytest 进行单元测试等；最后介绍了如何部署上线，包括使用自己搭建的服务器或使用各种云服务，以及如何

权衡和选择这些不同的解决方案

5.C语言下的int占多少字节？64位呢？

64位系统下C语言中int还是占4字节，32位，与32位系统中没有区别！

6.还有 django 报503 的错误 的时候 怎么查找错误，pycharm里DBUG有了解吗

7.还有 Linux 显示 所有磁盘 IO速率的 指令（不是top）

# 查看io进程

命令：iotop

# 查看当前磁盘IO 读写

sar -b 1 10

1、怎么判断session过期

(1).session其实就是一个Map，键=值对，通过session.getAttribute("name");获得session中设置的参数

(2).session的过期时间是从什么时候开始计算的？是从一登录就开始计算还是说从停止活动开始计算？

答：从session不活动的时候开始计算，如果session一直活动，session就总不会过期。

从该Session未被访问,开始计时; 一旦Session被访问,计时清0;

(3).设置session的失效时间

(4).request.getSeesion(boolean）方法，一下子让我恍然大悟。这个方法里面传了一个boolean值，这个值如果是true，那么如果当前的request的session不可用，那么就

创建新的会话，如果存在就返回当前的会话。如果参数是false，那么在request的当前会话不存在的时候就返回null。

2、怎么实现用户避免重复

3、购物车怎么和数据库交互实现增删改查

4、list去重的时间复杂度是多少

5、查看进程的Linux命令是什么

ps -aux

6、使用Linux命令查看django在内存中的占用量

7、git怎么返回上一个时间段

8、git怎么合并

1.git merge bugFix

2.git rebase

9、django、MySQL、redis默认端口号

8000 3306 6379

10、什么是进程，什么是线程

进程：是资源分配的最小单位，创建和销毁开销较大，程序文件的一次执行；

线程：是CPU调度的最小单位，开销小，切换速度快；

11、MySQL数据库怎么优化

(1)、选取最适用的字段属性

MySQL可以很好的支持大数据量的存取，但是一般说来，数据库中的表越小，在它上面执行的查询也就会越快。因此，在创建表的时候，为了获得更好的性能，我们可以将表

中字段的宽度设得尽可能小。

(2)、使用连接（JOIN）来代替子查询(Sub-Queries)

(3)、使用联合(UNION)来代替手动创建的临时表

(4)、事务

BEGIN; INSERT INTO salesinfo SET CustomerID=14; UPDATE inventory SET Quantity=11 WHERE item='book'; COMMIT;

事务的另一个重要作用是当多个用户同时使用相同的数据源时，它可以利用锁定数据库的方法来为用户提供一种安全的访问方式，这样可以保证用户的操作不被其它的用户所

干扰。

(5)、锁定表

LOCK TABLE inventory WRITE SELECT Quantity FROM inventory WHERE Item='book';

...

UPDATE inventory SET Quantity=11 WHERE Item='book'; UNLOCKTABLES

(6)、使用外键

(7)、使用索引

索引是提高数据库性能的常用方法，它可以令数据库服务器以比没有索引快得多的速度检索特定的行，尤其是在查询语句当中包含有MAX(),MIN()和ORDERBY这些命令的时候，

性能提高更为明显。

(8)、优化的查询语句

面试问题：

1、爬虫的实现流程？

1、要爬取的url

2、获取整个页面

3、用re、beautifulsoup、Xpath抓取需要的信息

4、写入本地文件

5、运用UA池、第三方代理、cookie池防止反爬

2、递归爬取页面怎么实现？

不会

3、同步异步的区别，异步IO？

同步：合作关系，一个事件完成会通知其他事件；异步，事件执行之间互不影响，(也有有影响的比如同时读写一个文件)

4、在项目中，进程池中有一个子进程死了，怎么查找出来？

通过查看子进程的PID

5、添加购物车PC端加入一个，如果在移动端显示？如果有两个机房，一个联通接口，一个电信接口，如果解决不同运营商的问题？

通过数据库实时刷新，第二问不会。

问题：说一下做这个项目做了多久？几个人做的？项目中有多少个py文件？

问题：用过git吧，git怎么创建分支？

git branch \*\*\*

问题：描述一下这个项目的实现流程？

问题：这个项目中哪里用到了进程，讲一下这些进程是怎么工作的？

问题：简单说一下redis

redis是一款优秀的nosql（非关系型），key-vlaue,内存级别的数据库！redis是单线程,操作是安全的！

关系型:用一张二维表格来表示和存储现实生活中对象的这一类数据库。 oracle ，mysql

非关系型（nosql）:不用二维表格，比如key-value ,文档等数据.Redis,MongoDB ,Hadoop

(1).Redis支持数据的持久化，会根据一定的策略持久化到磁盘,即使断电也不会丢失数据.

(2).Redis不仅仅支持简单的key-value类型的数据，同时还提供list，set，zset，hash等数据结构的存储。

(3).Redis支持数据的备份，即master-slave模式的数据备份

问题：装饰器有什么用？为什么要用到装饰器？

问题：Linux的常用命令都有哪些

ps,ls,pwd,find,mkdir

问题：都用过哪些框架？

Django，tornado

问题：Pycharm付费用的么？

问题：Mysql怎么做排序和分组？举个例子，说下sql语句。

select a.sort,a.gp from (

select \* from sort\_group order by sort desc

) a

group by a.gp

问题：mysql要查询前5条数据怎么写？

select \* from table\_name limit 0,5

问题：http和https的端口号分别是多少？

http:80

https:443

问题：看到你这个工作经历是到18年5月份，那你在达内这个是什么情况？

面试问题

1.介绍一下MySQL的事物

MySQL事务保证了数据库的完整性，可以使得批量的SQL语句，要么全部执行，要么全部不执行。

MySQL事务有四大特性（ACID）:

（一）原子性。（Atomicity）

事务的操作要么全部执行，要么全部不执行。若事务在操作的过程中发生什么错误，则事务将会回滚到事务开始前的状态，保证事务中的操作是不可分割的。

（二）一致性。（Consistency）

事务的操作前后，数据库的完整性没有被破坏。

（三）隔离性。（Isolation）

事务与事务之间的操作互相隔离。在多个CPU的环境下，多事务并发处理时，事务的隔离性可防止多个事务交叉对数据进行处理而导致数据不一致等情况的发生。隔离有

四种级别：(1)读未提交；（2）读已提交；（3）可重复读；（4）串行化。在稍后会详细介绍这四种隔离级别。

（四）持久性。（Durability）

事务开始和完成前后，对数据库的操作是持久的。即使是系统故障也不会丢失。

2.列表如何去重

(1)ids = [1,2,3,3,4,2,3,4,5,6,1]

news\_ids = []

for id in ids:

if id not in news\_ids:

news\_ids.append(id)

print news\_ids

(2)ids = [1,4,3,3,4,2,3,4,5,6,1]

ids = list(set(ids))

3.介绍一下你在项目中负责的模块

4.如何实现用户注册用户名不能重复

5.如何实现购物车加一减一操作

1、进程，线程的区别

进程：是资源分配的最小单位，创建和销毁开销较大；

线程：是CPU调度的最小单位，开销小，切换速度快；

（1）多进程中同一个变量各自有一份拷贝在每个进程中，互不影响；

（2）多线程中，所有变量都由所有线程共享，任何一个变量都可被任何一个线程修改。线程之间共享数据的最大危险在于多个线程同时更改一个变量，把内容改乱。

2、用户重复提交订单，后台怎么处理

3、描述登录流程

4、注册时用户名重复怎么处理

5、git 中项目两个版本的差别，（命令）

6、什么是装饰器

7、mysql的级联查询，

8、描述下tornado的使用方法

9、linux 安装django的方法

10、验证码随机生成数字调用那个方法，生成图片调用那个方法

生成器以及迭代器的定义以及实现，http如何实现请求（报文方面去讲解） 中间件如何工作的 介绍的工作（在介绍工作的时候会随机的问相关的例如：我说使用MySQL

，他会打断我，让我回答MySQL的索引有哪些，作用，建立索引的依据）