面试准备

110道python面试真题：[https://zhuanlan.zhihu.com/p/54430650](https://zhuanlan.zhihu.com/p/54430650" \t "_blank)

后端面试题总汇：[https://github.com/yongxinz/back-end-interview](https://github.com/yongxinz/back-end-interview" \t "_blank)

【python\_developer】:<https://github.com/xiandong79/Python_Developer>

【Django面试题】：<https://www.cnblogs.com/chongdongxiaoyu/p/9403399.html>

【Python（推荐）】:<https://github.com/CyC2018/CS-Notes>

【Python】:<https://github.com/taizilongxu/interview_python>

**算法思想**

排序（七种）：<https://www.cnblogs.com/ltchu/p/6610576.html>

动态规划：**<https://www.cnblogs.com/xiaoshen666/p/11014954.html>**

双指针：**<https://blog.csdn.net/weixin_42552135/article/details/82556234>**

贪心算法：**用贪心算法只能通过解局部最优解的策略来达到全局最优解。**

[**https://blog.csdn.net/a925907195/article/details/41314549**](https://blog.csdn.net/a925907195/article/details/41314549)

[**https://www.cnblogs.com/gavanwanggw/p/7141358.html**](https://www.cnblogs.com/gavanwanggw/p/7141358.html)

分治：**<https://blog.csdn.net/qq_25740691/article/details/78894177>**

（1）分解，将要解决的问题划分成若干规模较小的同类问题；

（2）求解，当子问题划分得足够小时，用较简单的方法解决；

（3）合并，按原问题的要求，将子问题的解逐层合并构成原问题的解。

二分查找：**<https://blog.csdn.net/Abysscarry/article/details/87388195>**

搜索：**<https://blog.csdn.net/qq_42472682/article/details/97512878>**

**深度优先搜索**

**广度优先搜索**

**基础篇**

1. **问我对哪个语言比较熟悉，问了我python和C++、JAVA的区别？**

<https://blog.csdn.net/qq_40518671/article/details/89151738>

**可变数据类型：列表（list）、字典（dict）、集合（set）；**

**不可变数据类型： 数值型（int）、字符串型（string）、元组（tuple）**

1. **Python代码执行原理？**

<https://www.seoxiehui.cn/article-152715-1.html>

**我们运行python文件程序的时候，python解释器将源代码转换为字节码，然后再由python解释器来执行这些字节码。**

1. **解释Python的对象？**

<https://www.cnblogs.com/harvyxu/p/8535930.html>

面向对象是一种基于结构分析的，以数据为中心的程序设计思想。

1. **python中字典的底层是怎么实现的？**

**dict的底层是依靠哈希表(Hash Table)进行实现的，使用开放地址法解决冲突**.。

**哈希表**

**哈希表是key-value类型的数据结构，通过关键码值直接进行访问。通过散列函数进行键和数组的下标映射从而决定该键值应该放在哪个位置。**

1. **谈谈super原理？**

<https://blog.csdn.net/qq_26442553/article/details/81775449>

**1、 super().\_\_init\_\_相对于类名.\_\_init\_\_，在单继承上用法基本无差**

**2、但在多继承上有区别，super方法能保证每个父类的方法只会执行一次，而使用类名的方法会导致方法被执行多次，具体看前面的输出结果**

**3、多继承时，使用super方法，对父类的传参数，应该是由于python中super的算法导致的原因，必须把参数全部传递，否则会报错**

**4、单继承时，使用super方法，则不能全部传递，只能传父类方法所需的参数，否则会报错**

**5、多继承时，相对于使用类名.\_\_init\_\_方法，要把每个父类全部写一遍, 而使用super方法，只需写一句话便执行了全部父类的方法，这也是为何多继承需要全部传参的一个原因**

1. **不同的父类**中存在 **同名的方法**，**子类对象**在调用方法时，会调用**哪一个父类中**的方法呢**？**

**Python 中的 MRO —— 方法搜索顺序**

* **Python 中针对 类 提供了一个内置属性 \_\_mro\_\_ 可以查看方法搜索顺序。**
* **MRO 是 method resolution order，主要用于在多继承时判断 方法、属性 的调用 路径。**

1. **for...in...循环的本质？**

**【答案】：for item in Iterable 循环的本质就是先通过iter()函数获取可迭代对象Iterable的迭代器，然后对获取到的迭代器不断调用next()方法来获取下一个值并将其赋值给item，当遇到StopIteration的异常后循环结束。**

1. **简述面向对象中\_\_new\_\_和\_\_init\_\_区别。**

<https://zhuanlan.zhihu.com/p/54430650>

1. **\_\_new\_\_是一个静态方法,而\_\_init\_\_是一个实例方法.**
2. **\_\_new\_\_方法会返回一个创建的实例,而\_\_init\_\_什么都不返回.**
3. **只有在\_\_new\_\_返回一个cls的实例时后面的\_\_init\_\_才能被调用.**
4. **当创建一个新实例时调用\_\_new\_\_,初始化一个实例时用\_\_init\_\_.**

**分别使用\_\_metaclass\_\_,\_\_new\_\_和\_\_init\_\_来分别在类创建,实例创建和实例初始化**

1. **python的垃圾回收机制（三种方法以及循环引用如何解决）？**

**Python GC 主要使用引用计数（reference counting）来跟踪和回收垃圾。**

**在引用计数的基础上，通过“标记-清除”（mark and sweep）解决容器对象可能产生的循环引用问题。**

**通过“分代回收”（generation collection）以空间换时间的方法提高垃圾回收效率。**

**1 引用计数**

**PyObject是每个对象必有的内容，其中ob\_refcnt就是做为引用计数。当一个对象有新的引用时，它的ob\_refcnt就会增加，当引用它的对象被删除，它的ob\_refcnt就会减少.引用计数为0时，该对象生命就结束了。**

**2 标记-清除机制**

**基本思路是先按需分配，等到没有空闲内存的时候从寄存器和程序栈上的引用出发，遍历以对象为节点、以引用为边构成的图，把所有可以访问到的对象打上标记，然后清扫一遍内存空间，把所有没标记的对象释放。**

**3 分代技术**

**分代回收的整体思想是：将系统中的所有内存块根据其存活时间划分为不同的集合，每个集合就成为一个“代”，垃圾收集频率随着“代”的存活时间的增大而减小，存活时间通常利用经过几次垃圾回收来度量。**

1. **python的生成器和迭代器？**

**迭代器：**<https://www.cnblogs.com/wangcoo/p/10018363.html>

<https://www.cnblogs.com/wj-1314/p/8490822.html>

**next()函数调用并不断返回下一个值得对象称为迭代器 (Iterator)**

1. **python里装饰器的作用，本质是什么，怎么去构造一个装饰器？**

**装饰器：装饰器其实就是一个闭包，把一个函数当做参数后返回一个替代版函数，闭包是装饰器的核心。**

**目的是在不改变原函数的情况下，给被修饰函数增加新的功能。**

**闭包：**

**在一个外函数中定义了一个内函数，内函数里运用了外函数的临时变量，并且外函数的返回值是内函数的引用。这样就构成了一个闭包。**

1. **数据结构链表、栈、队列、堆的原理以及大体实现过程？**

<https://www.cnblogs.com/hedeyong/p/7841548.html>

1. **数组和链表的区别，优缺点？**

<https://blog.csdn.net/weibo1230123/article/details/82011889>

**数组的查询时间复杂度O(1) ，修改的时间复杂度O(n)**

**链表的查询时间复杂度O(n) ，修改的时间复杂度O(1)**

1. **python2和python3区别？列举5个**

**1、Python3 使用 print 必须要以小括号包裹打印内容，比如 print('hi')。Python2 既可以使用带小括号的方式，也可以使用一个空格来分隔打印内容，比 如 print 'hi'**

**2、python2 range(1,10)返回列表，python3中返回迭代器，节约内存**

**3、python2中使用ascii编码，python中使用utf-8编码**

**4、python2中unicode表示字符串序列，str表示字节序列；python3中str表示字符串序列，byte表示字节序列；**

**5、python2中为正常显示中文，引入coding声明，python3中不需要**

**6、python2中是raw\_input()函数，python3中是input()函数**

1. **解释一下python中的GIL是什么？GIL是单线程的，那么python中多线程的实现有什么用。**

**GIL 是python的全局解释器锁，同一进程中假如有多个线程运行，一个线程在运行python程序的时候会霸占python解释器（加了一把锁即GIL），使该进程内的其他线程无法运行，等该线程运行完后其他线程才能运行。如果线程运行过程中遇到耗时操作，则解释器锁解开，使其他线程运行。所以在多线程中，线程的运行仍是有先后顺序的，并不是同时进行。**

1. **分布式锁的实现（Redis实现分布式锁）？、**

**基于[数据库](http://lib.csdn.net/base/mysql" \o "MySQL知识库" \t "_blank)实现分布式锁**

依赖数据库的一张表，一种是通过表中的记录的存在情况确定当前是否有锁存在，另外一种是通过数据库的排他锁来实现分布式锁。 **基于缓存（[Redis](http://lib.csdn.net/base/redis" \o "Redis知识库" \t "_blank)，memcached，tair）实现分布式锁  
基于Zookeeper实现分布式锁**

<https://www.cnblogs.com/lice-blog/p/11616429.html>

1. **python中的map是怎么实现的，知道java中的hashmap底层是怎么实现的吗？**

<https://blog.csdn.net/suifeng629/article/details/82179996>

1. **关于快排、堆排序、冒泡排序、选择排序等七种算法的掌握。**

<https://www.cnblogs.com/ltchu/p/6610576.html>

1. **深拷贝和浅拷贝的区别？**

**深拷贝：复制了原来的对象，修改深拷贝的值或者改变原对象的值，另一对象的值不改变；**

**浅拷贝：只是多了一个引用，改变原对象的值，则浅拷贝对象的值改变。**

1. **进程和线程？两个的区别？**

* **进程：**

**1、操作系统进行资源分配和调度的基本单位，进程之间相互独立。  
2、稳定性好，如果一个进程崩溃，不影响其他进程，但是进程消耗资源大，开启的进程数量有限制**

* **线程  
  1、CPU进行资源分配和调度的基本单位，线程是进程的一部分，是比进程更小的能独立运行的基本单位，一个进程下的多个线程可以共享该进程的所有资源  
  2、如果IO操作密集，则可以多线程运行效率高，缺点是如果一个线程崩溃，都会造成进程的崩溃**
* **应用  
  IO密集的用多线程，在用户输入，sleep 时候，可以切换到其他线程执行，减少等待的时间  
  CPU密集的用多进程，因为假如IO操作少，用多线程的话，因为线程共享一个全局解释器锁，当前运行的线程会霸占GIL，其他线程没有GIL，就不能充分利用多核CPU的优势。**

1. **线程间的通信方式**

**线程间通信的模型有两种：共享内存和消息传递**

**1、基于 volatile 关键字**

来实现线程间相互通信是使用共享内存的思想，大致意思就是多个线程同时监听一个变量，当这个变量发生变化的时候 ，线程能够感知并执行相应的业务。

#### ****2、使用Object类的wait() 和 notify() 方法****

#### ****3、使用JUC工具类 CountDownLatch****

#### ****4、使用 ReentrantLock 结合 Condition****

#### ****5、LockSupport实现线程间的阻塞和唤醒****

**网络篇**

1. **TCP具体是通过怎样的方式来保证数据的顺序化传输、可靠传输呢？**

* **顺序化传输**  
  **偏移量（依靠，IP地址中的字节序列）**
* **可靠传输**

**超时重传：如果一个已经发送的报文段在超时时间内没有收到确认，那么就重传这个报文段。**

1. **HTTPs和HTTP的区别，HTTPS实现的原理过程，SSL加密，非对称加密和对称加密在HTTPS中是怎么用到的。**

<https://www.cnblogs.com/jesse131/p/9080925.html>

1. **http有哪些版本，1.0和1.1有什么区别？**

<https://blog.csdn.net/elifefly/article/details/3964766>

1. **cookie和session了解吗？**

这篇写的非常好：<https://www.cnblogs.com/wswang/p/6062461.html>

* **Cookies是服务器在本地机器上存储的小段文本并随每一个请求发送至同一个服务器，是一种在客户端保持状态的方案。**
* **Cookie是访问某些网站以后在本地存储的一些与网站相关的信息**

### 创建过程

### ****服务器发送的响应报文包含 Set-Cookie 首部字段，客户端得到响应报文后把 Cookie 内容保存到浏览器中。****

### ****客户端之后对同一个服务器发送请求时，会从浏览器中取出 Cookie 信息并通过 Cookie 请求首部字段发送给服务器。****

* 会话状态管理（如用户登录状态、购物车、游戏分数或其它需要记录的信息）
* 个性化设置（如用户自定义设置、主题等）
* 浏览器行为跟踪（如跟踪分析用户行为等）

### Session

**Session是存在服务器的一种用来存放用户数据的类HashTable结构。**

当浏览器 第一次发送请求时，服务器自动生成了一个HashTable和一个Session ID用来唯一标识这个HashTable，并将其通过响应发送到浏览器。当浏览器第二次发送请求，会将前一次服务器响应中的Session ID放在请求中一并发送到服务器上，服务器从请求中提取出Session ID，并和保存的所有Session ID进行对比，找到这个用户对应的HashTable。

**一个在客户端一个在服务端**

1. **Socket了解吗？**

**socket通常也称作"套接字"，用于描述IP地址和端口，是一个通信链的句柄，应用程序通常通过"套接字"向网络发出请求或者应答网络请求。**

**socket可以实现在不同的计算机之间传输数据，也就是网络传输数据。**

<https://blog.csdn.net/weixin_39634961/article/details/80236161>

<http://www.360doc.com/content/11/0609/15/5482098_122692444.shtml>

1. **HTTP状态码，都说说有哪些吧？**
2. **数据链路层三大特性？**

**封装成帧，透明传输，差错检验（循环冗余校验(CRC)）。**

1. **Get 和Post 的区别？**
   1. **1. GET使用URL或Cookie传参。而POST将数据放在 请求头 中。**
   2. **2. GET的URL会有长度上的限制（没有明确规定，但是不同浏览器有不同的限制），则POST的数据则可以非常大。（小数据量:GET 大数据量：POST）**
   3. **3. POST比GET安全，因为数据在地址栏上不可见。**
2. **Python的IO多路复用是怎么实现的？**

<https://blog.csdn.net/bird73/article/details/79793308>

1. **并发与并行？**

**并发：（多线程就是并发），多个事件轮流交替执行。（吃一口饭，喝一口水）**

**并行：（多进程是并行），应用能够同时执行不同的任务。（吃饭并打电话）**

<https://blog.csdn.net/u013945548/article/details/50988456/>

1. **同步与异步？**

**同步是指：当程序1调用程序2时，程序1停下不动，直到程序2完成回到程序1来，程序1才继续执行下去。 （停下等待）**

**异步是指：当程序1调用程序2时，程序1径自继续自己的下一个动作，不受程序2的的影响。（不等待）**

**-----------------------------------------------------------------------------**

**同步是指：发送方发出数据后，等接收方发回响应以后才发下一个数据包的通讯方式。**

**异步是指：发送方发出数据后，不等接收方发回响应，接着发送下个数据包的通讯方式。**

1. **域名解析的过程？**
2. **打开了一个浏览器，输入[www.baidu.com](http://www.baidu.com/" \t "_blank)发生了些什么过程，中间问到了ARP的过程？**

**事件顺序：**

1. **浏览器获取输入的域名**[**www.baidu.com**](http://www.baidu.com/)
2. **浏览器向DNS请求解析（域名）www.baidu.com的IP地址**
3. **域名系统DNS解析出百度服务器的IP地址**
4. **浏览器与该服务器建立TCP连接(默认端口号80)**
5. **浏览器发出HTTP请求，请求百度首页**
6. **服务器通过HTTP响应把首页文件发送给浏览器**
7. **TCP连接释放。**
8. **浏览器将首页文件进行解析，并将Web页显示给用户。**

**框架篇**

【Django面试题】：<https://www.cnblogs.com/chongdongxiaoyu/p/9403399.html>

1. django框架和flask框架的比较
2. django中中间件的用法，以及执行流程
3. …

**数据库篇**

**【MySQL面试题】：**<https://github.com/yongxinz/back-end-interview/tree/master/MySQL>

1. **数据库的事务，四种隔离机制，以及产生的问题？**

**事务以及其属性：**

**（1）、原子性：事务是SQL语句的基本逻辑运行单元，其对数据的修改，要么全部执行，要么全部不执行。（要么都发生，要么都不发生。）**

**（2）、一致性：事务从一个状态改变到另外一个状态，事务开始和完成时，数据必须保持一致状态。（所有相关的数据规则都应用于事务的修改，来保持数据的完整性；）**

**（3）、隔离性：数据库提供一定的隔离机制（事物之间互不干扰）。（保证事务在不受外部并发的影响下能独立执行。）**

**（4）、持久性：事务完成后，它对数据的修改是永久性的。**

1. **redis的数据类型，以及这五种类型使用的场景**

**redis是一个完全开源免费的 key-value 内存数据库。**

**五种类型：strings、hash、list、sets、sorted sets。**

1. **为什么要用redis？ 好处在哪？**

**读写速度快；持久化；自动操作；快速的主--从复制、Sharding 技术。**

1. **常用的mysql引擎有哪些?各引擎间有什么区别?**

<https://blog.csdn.net/qq_32483145/article/details/80191323>

1. **数据库索引的三种常用方式？**

**【磁盘I/O操作次数等于索引树的高度】：相比较于IO操作，内存的比较耗时可以忽略不计。所以：IO次数越少，查找性能越优。**

**B树：**

**1.根结点至少有两个子女。**

**2.每个中间节点都包含k-1个元素和k个孩子，其中 m/2 <= k <= m**

**3.每一个叶子节点都包含k-1个元素，其中 m/2 <= k <= m**

**4.所有的叶子结点都位于同一层。**

**5.每个节点中的元素从小到大排列，节点当中k-1个元素正好是k个孩子包含的元素的值域分划。**

**用途：**

**主要用于文件系统以及部分数据库索引（比如：MongoDB菲关系型数据库）**

**B+ 树：**

**1.有k个子树的中间节点包含有k个元素（B树中是k-1个元素），每个元素不保存数据，只用来索引，所有数据都保存在叶子节点。**

**2.所有的叶子结点中包含了全部元素的信息，及指向含这些元素记录的指针，且叶子结点本身依关键字的大小自小而大顺序链接（每一个叶子节点都带有指向下一节点的指针）。**

**3.所有的中间节点元素都同时存在于子节点，在子节点元素中是最大（或最小）元素。（最大元素在根节点）**

**【总结】：优于B树的几点：**

**1、【IO次数更少】单一节点存储更多的元素，相同数据量时，比B树“矮胖”（查询时，IO操作少，性能更优）**

**2、【查询性能稳定】 B+ 树每一次查找都到叶子节点，性能稳定；B树最好情况为只查根节点，最坏为查到叶子节点。**

**3、【范围查询简便】** 叶子节点形成有序链表，**B+树范围查询时只需在链表上遍历即可。**

1. **数据库联接操作，左连接，右链接，全链接的操作以及区别，问了数据库的三大范式的理解。**
2. …

**Linux篇**

1. linux命令，查内存，查cpu，查进程
2. linux和windows有什么区别
3. linux的GUI和windows的GUI有什么区别吗？
4. linux的GUI定层原理是什么，怎么实现的？
5. **..**

**立个Flag**：学会知识点，找到好实习！

待续…