第一部分，技术理论面试题。

1， 什么是魔法函数?

2， 什么是闭包?

3， 深拷贝和浅拷贝的区别，并举例说明。

4， 如何在一个函数内部修改全局变量?

5， 什么是单例模式?

6， python中如何实现多线程?

7， Flask和diango的区别有哪些?

8， Pytest框架里面断言有哪些类型?

9， Is和==有什么区别?

10，装饰器有什么作用?

第二部分，架构中间件性能面试题。

1， 常用的中间件有哪些？如何测试它们?

2， 集群和分布式有什么区别?

3， 幂等实现原理是什么?

4， 说说Redis有哪些特性，哪些地方容易出问题?

5， 数据库锁有哪些类型，什么情况下需要这些锁?

6， 数据库索引有哪些优点和缺点?

第三部分，笔试部分，手撸代码。

1， 手写一个装饰器

2， 写一个函数，实现随机一个数组，长度和元素不固定，输出任意两个元素相加结果为N的方法

3，给定一个字符串，判断其是否合法。说明：字符串内包含“(,),{,},[,]"对象，随机位置，检查同类型括号的对称情况，如果存在左右对称括号中间只有单独一个括号的字符串，即为不合法。

4， 一个数组内有随机的10整数，随机取三个元素，他们的和为0，写个方法函数，返回所有的组合情况，要去除重复的组合内容

5， 写一个函数方法，实现输入两个日期，输出两个日期间隔的天数

6， 随机一个数组，求其中两个元素乘积最大的组合。

第四部分，系统设计

1，请设计一个资产管理系统，简述功能清单和逻辑，还有技术选型。

上述的面试题大家可以先尝试看看能否很好的回答出来，当然很多题目没有唯一答案，主要考察思路这块吧。

下面是我对面试题的理解和解答，主要是freestyle的想法，仅供参考，也欢迎大家给出更深层次的思考，文案底部留言即可。

第一部分，第一题什么是魔法函数?

魔法函数，像“__init__”这样的构造函数就是魔法函数，平时也经常用。魔法函数的作用是给类增加一些额外的功能，这道题目是一道理论题，大家可以自行百度。

第一部分，第二题什么是闭包?

闭包，通俗点讲就是内部函数调用了外部函数的变量，这样就形成了一个闭包。需要对函数作用域和函数生命周期有所了解，例如，我们不能直接调用内部函数，外部函数如果不创建，直接调用内部函数会抛错。如何解决闭包的问题呢？如果是变量，那么通过global把变量设定为全局变量，如果是函数，我们可以return。

第一部分，第三题深拷贝和浅拷贝的区别，并举例说明。

这题考到python里面数据存储概念，我们声明赋值变量a=123，然后输出id(a)会打印一串数字，这串数字就是变量a在内存的存储位置。这个时候，如果在来一个b=123，输出id(b)，打印出来的结果和id(a)是一样的，如果a变了，b也跟着一起变，这个就是浅拷贝；反过来，深拷贝就是两个对象内容数据相同，但是在内存的位置不一样，Python里面的copy函数方法，就是深拷贝。

第一部分，第四题如何在一个函数内部修改全局变量

这一题在上面第二题，谈闭包问题的解决办法有提到，用global关键词就可以实现。

第一部分，第五题什么是单例模式?

单例模式是设计模式的一种，Python中，通过__new__方法创建对象，再通过__init__方法创建内存空间，单例模式要求，类创建的对象，在内存中只有唯一的一个实例，且每次实例化生成的对象，内存空间地址都是相同的。举个例子，系统给打印机发送打印多个文件指令，打印机最多只能同时打印一个文件，多个文件只能多次打印。

第一部分，第六题python中如何实现多线程?

线程分为内核级线程和用户级线程，而python因为无法访问内核内存空间，只能依赖生成器、greenlets和类似的库实现用户级线程，Python中多线程一般用threading模块实现多线程。

第一部分，第七题Flask和Diango的区别有哪些?

Flask和Diango的区别，类似于java编辑器eclipse和myeclipse。Diango自身已经集成了很多的方法和类库，所以很重，相对Flask灵活度和自由度不够高；Flask是轻量级框架，更多的需要开发者按照自己的需要去扩展，两者的生态都很丰富，但是性能方面，Flask较Diango要好一些。

第一部分，第八题pytest框架里断言有哪些类型?

Pytest是unittest的升级，较unittest断言语法要更简洁一些，pytest的扩展库pytest-assume支持多重断言。一般用到的断言有状态码断言，响应时长断言，返回内容解析对比断言，数据库查询对比断言。

第一部分，第九题is和==有什么区别?

Is,比较的是两个对象是否指向同一个内存地址，也就是两个对象是否为同一个实例对象；==比较的是两个对象内容和数据类型是否一样，默认调用对象的__eq__()方法

第一部分，第十题装饰器有什么作用?

装饰器是给某个函数（类）增加一些功能，但不影响函数（类）自身的函数或类。装饰器有很多种，可以分为类装饰器和函数装饰器两大类，能够帮助我们简化代码和方便维护。

第二部分，第一题常用的中间件有哪些？如何测试它们?

中间件是一种独立的系统软件或服务程序，作用是为处于自己上层的应用软件提供运行与开发的环境，帮助用户灵活、高效地开发和集成复杂的应用软件。常见的中间件有mq消息队列，xxl-job任务调度中心，阿波罗配置中心，Redis等。对于中间件这块，我没有做一些专项测试工作，除了按照业务功能需求去设计一些边界值，等价类之类的测试方法外，做过一些简单的压力和疲劳测试。需要注意一点，定时任务调度中心和关联任务的服务所在服务器时间要保持一致。

第二部分，第二题集群和分布式有什么区别?

通常来讲，集群内的服务，提供的功能都是一样的；而分布式里面，每个服务提供的功能可能不一样。集群强调集群内服务器位置集中，方便统一管理；分布式不要求位置，只要网络连通即可。集群是一种物理形态，分布式是一种工作方式。（回答的时候，只说了第一点，没有这么完整）

第二部分，第三题幂等实现原理是什么?

这一题切入点是问我，支付的时候，怎么保证不会重复支付一笔订单。我回答通过幂等实现，然后面试官追问我，怎么样实现幂等，它的原理是什么？很可惜，这后面的我就答不上来了，查了资料明白，防重复支付有几种办法，第一种，把下单和支付放在一个事务里面处理，要么全成功，要么全失败；第二种，就是幂等，幂等实现办法有好几种，token机制，悲观锁，乐观锁，分布式锁，状态机幂等。关于token机制，采用 Token+Redis（Redis 是单线程的，处理需要排队）的解决方案。处理的流程是，在数据提交前要向服务器申请带有有效时间的 Token，然后 Token 放到 Redis 或 JVM 内存中，当数据正式提交到后台要校验 Token 并删除 Token。其余的几种有兴趣的可以再查一下资料了解一下。

第二部分，第四题说说Redis有哪些特性，哪些地方容易出现问题?

Redis是基于内存的高性能键值对数据库，并且支持主从模式（读写分离）；Redis是单线程的，通过队列技术进行串行访问，消除传统数据库串行控制的开销；支持丰富的数据类型，支持事务，操作都是原子性（对数据的更改要么全部执行，要不全部不执行），支持设置数据过期时间，有自己的回收策略。以上是redis的一些特性举例。Redis在如下一些地方容易出现问题：首先，redis存储在内存，断电易失，大部分业务场景需要涉及到增量和全量更新，容易出现数据丢失的问题；redis是key-value管理数据，当key很多，有一些类似的key等情况时，容易出现更新混乱（代码层面），再就是并发更新同一个key；缓存击穿，请求redis不存在的数据，压力转移到关系型数据库，导致关系型数据库异常，应对缓存击穿防止策略有互斥锁、异步更新、拦截机制；缓存雪崩，因redis
失效策略配置问题，同一时间缓存大面积失效，未更新数据同步到redis之前，请求压力堆积到关系型数据库，导致数据库异常，防范策略有失效策略多检查，随机值或通过规则防止同一时间集体失效，使用互斥锁（降低性能），双缓存（双主或主从）。结合实际的业务和系统设计也会有一些需要多关注的地方。

第二部分，第五题数据库锁有哪些类型，什么情况下需要这些锁?

数据库使用到锁，是为了保证数据的一致性。拿mysql举例，用到三种类型（级别）锁定机制：表级锁定，行级锁定和页级锁定；细分还有共享锁，排它锁，意向锁等。一般跟进实际业务场景影响数据范围，来选择使用哪种级别的锁定机制，比如，只做单条数据的更新，就建议使用行级锁。行级锁下面共享锁适用于：用来确认某行记录是否存在，并确保没有人对这个记录进行UPDATE或者DELETE操作，如果当前事务也需要对该记录进行更新操作，则很有可能造成死锁。排他锁适用于：锁定行记录后需要进行更新操作的应用。

第二部分，第六题数据库索引有哪些优点和缺点?

先说优点：唯一索引可以保证每行数据的唯一性，其次可以提升数据查询速度（单表，表关联，分组排序）缺点：每次新增数据创建索引要耗费时间，降低了写数据的性能；索引需要占用物理空间；创建修改删除操作，索引也需要动态维护，拉低性能。

第三部分，第一题手写一个装饰器。

装饰器种类有很多，并没有制定装饰器类型，就随便写一个吧，这里就不贴代码了。

第三部分，第二题写一个函数，实现随机一个数组，长度和元素不固定，输出任意两个元素相加结果为N的方法

分析：N为固定值，可以从数组中循环取出一个数a，然后用N-a得到第二个匹配元素b，最后查找b是否在剩下的数组元素里面；用lambda表达式，应该也可以实现。

第三部分，第三题给定一个字符串，判断其是否合法。

说明：字符串内包含“(,),{,},[,]"对象，随机位置，检查同类型括号的对称情况，如果存在左右对称括号中间只有单独一个括号的字符串，即为不合法（Ps:需要左右括号匹配类型也一致才算合法）

分析：首先过滤掉字符串其他元素，保留“(){}[]”元素对象，并且位置顺序不变。然后通过从左往右的顺序找到最后一个左括号（也可以从右往左，找最后一个右括号），然后继续往右进行逐一匹配右括号，遇到一个不匹配的即为不合法。我回答的不是这个版本，我忽略了顺序，这里运用到堆栈概念。

第三部分，第四题一个数组内有随机的10整数，随机取三个元素，他们的和为0，写个方法函数，返回所有的组合情况，要去除重复的组合内容

分析：和第二题类似，增加了一个元素求和，需要输出组合情况，并去重。我们可以用加法或者用减法进行倒推，三层循环，每层取出一个元素并去掉已取出的元素，然后拿符合条件的三个元素组成一个数组，排序，检查一下是否已加入结果数组，未加入再append，然后把结果return。下面是用减法的示例：

第三部分，第五题写一个函数方法，输入两个日期(yyyymmdd)，输出两个日期相差多少天。

分析：这个比较简单，使用python的datetime模块即可。

第三部分，第六题随机一个数组，求其中两个元素乘积最大的组合。

分析，既然需要求最大乘积组合，找到数组里面最大的两个元素相乘就可以了。

第四部分，第一题请设计一个资产管理系统，简述功能清单和逻辑，还有技术选型。

这道题综合考量个人软件设计能力，我们拆成两个步骤来。第一步，先进行需求分析，然后进行功能设计，给出功能清单；第二步，技术选型系统设计。

，前端后端数据库服务器等。

先看功能清单：

1， 登录，用户权限管理

2， 设备列表，设备详情

3， 资产设备登记/启用/修改信息

4， 资产设备借出/归还

5， 资产设备报修/报失

6， 资产盘点记录

7， 用户操作日志记录

按照功能清单进一步分析：登录是接公司内部统一登录还是自己做一套用户管理模块（需要按照实际情况调研），需要有完善的资产设备的状态机流转流程（草稿、未占用、占用中、已报修、已报失、等）

接下来进行技术选型和系统设计：

因为这个资产管理系统会有许多关联查询的情况，其次对于资产管理，数据完整性很重要，所以我们选取关系型数据库mysql。对于资产管理系统并不会出现大并发的情况，而且数据量也不会特别大，mysql单表在百万级性能都还ok。前端因为我对vue之类的框架不太熟练，然后资产管理系统除了在B端用，最好也可以适配C端浏览器，Bootstrap是个不错的选择。后端就用python的flask了，轻量级，生态也比较全面（其实我就会flask，diango我不会，哈哈），用户管理登录这块，按照传统来讲应该是要接统一登录，如果自己写，身份鉴权就需要用到redis更合适一些。技术选型就这些，最后就是系统设计了。首先数据表设计，资产详情表、用户角色表、用户信息表、资产盘点信息表、用户操作记录表。资产详情表给资产类型，名称等字段增加索引（5个以内），用户操作记录表按照月份取模分表（这个其实没太大必要）。另外，数据库表设计要注意命名规范，表必须要有主键。接着进行接口设计，按照功能清单逐一实现，需要注意接口可扩展性，后面可能会更新版本。

至于最后服务部署，就uWSGI+Nginx+Flask在Linux下的部署吧，代码管理和发布用git和Jenkins工具。系统设计肯定不是这么简单就完事了的，这里只能讲个大概，很多地方都需要进一步的扩展实现，大家可以自己再丰富或修改一下，哈哈。

感慨一下，测试这块越往后面，涉及的范围越来越广，要求深度也越来越高，因为本身这个行业的技术就在不停的升级，想要做好就需要不断的学习和创新。不知道有没有很多小伙伴会感觉到了一个发展提升的瓶颈：广度够了，深度不够，需要选一个业务域或者是技术栈进行深耕；其次是自己适合做管理还是适合做技术的困惑，有些时候兴趣重要，天赋也很重要