**Python面试题**

**1.python的生成器，迭代器，装饰器分别解释一下**

1）迭代器是一个更抽象的概念，任何对象，如果它的类有next方法和iter方法返回自己本身。对于string、list、dict、tuple等这类容器对象，使用for循环遍历是很方便的。在后台for语句对容器对象调用iter()函数，iter()是python的内置函数。iter()会返回一个定义了next()方法的迭代器对象，它在容器中逐个访问容器内元素，next()也是python的内置函数。在没有后续元素时，next()会抛出一个StopIteration异常

装饰器的作用和功能：

引入日志

函数执行时间统计

执行函数前预备处理

执行函数后的清理功能

权限校验等场景

缓存

2）生成器（Generator）是创建迭代器的简单而强大的工具。它们写起来就像是正规的函数，只是在需要返回数据的时候使用yield语句。每次next()被调用时，生成器会返回它脱离的位置（它记忆语句最后一次执行的位置和所有的数据值）

区别：生成器能做到迭代器能做的所有事,而且因为自动创建了\_\_iter\_\_()和next()方法,生成器显得特别简洁,而且生成器也是高效的，使用生成器表达式取代列表解析可以同时节省内存。除了创建和保存程序状态的自动方法,当发生器终结时,还会自动抛出StopIteration异常

**2.for循环与迭代器有什么区别**

通过 for…in… 循环依次拿到数据进行使用的过程称为遍历，也叫迭代。我们把可以通过 for…in… 语句迭代读取数据的对象称之为可迭代对象。

for…in… 循环的本质：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 （for…in..会自动处理 StopIteration 异常）

迭代器是一个实现了迭代器协议的对象，Python中的迭代器协议就是有next方法的对象会前进到下一结果，而在一系列结果的末尾是，则会引发StopIteration。

**3.浅拷贝与深拷贝有什么区别**

在Python中对象的赋值其实就是对象的引用。当创建一个对象，把它赋值给另一个变量的时候，python并没有拷贝这个对象，只是拷贝了这个对象的引用而已。

浅拷贝：拷贝了最外围的对象本身，内部的元素都只是拷贝了一个引用而已。也就是，把对象复制一遍，但是该对象中引用的其他对象我不复制

深拷贝：外围和内部元素都进行了拷贝对象本身，而不是引用。也就是，把对象复制一遍，并且该对象中引用的其他对象我也复制。

1，深浅拷贝都是对源对象的复制，占用不同的内存空间。

2，不可变类型的对象，对于深浅拷贝毫无影响，最终的地址值和值都是相等的。

3，可变类型：

=浅拷贝： 值相等，地址相等

copy浅拷贝：值相等，地址不相等

deepcopy深拷贝：值相等，地址不相等

**4.Django的中间件有哪些。**

中间件是Django请求/响应处理的钩子框架。这是一个轻量级的低级“插件”系统，用于在全球范围内改变Django的输入或输出。每个中间件组件都负责完成一些特定的功能。例如，Django包含一个中间件组件 AuthenticationMiddleware，它使用会话将用户与请求相关联。

常用两中间件process\_request & process\_response

**5.请简述一下你做的这个项目的功能与你在当中做了哪些工作**

**6.请简述一下你印象最深的一个脚本以及如何实现它的**

**7.ansible模块你们是如何实现自动化批量部署的（如何一键执行命令）**

ansible是新出现的自动化运维工具，基于Python开发，集合了众多运维工具（puppet、cfengine、chef、func、fabric）的优点，实现了批量系统配置、批量程序部署、批量运行命令等功能。

ansible是基于模块工作的，本身没有批量部署的能力。真正具有批量部署的是ansible所运行的模块，ansible只是提供一种框架。主要包括：

(1)连接插件connection plugins：负责和被监控端实现通信；

(2)host inventory：指定操作的主机，是一个配置文件里面定义监控的主机；

(3)各种模块核心模块、command模块、自定义模块；

(4)借助于插件完成记录日志邮件等功能；

(5)playbook：剧本执行多个任务时，非必需可以让节点一次性运行多个任务。

首先我们得配置本地yum源，ansible安装成功之后，接下来进行Ansible的配置

**8.ansible如何实现动态主机识别的**

ansible，默认是调用/etc/ansible/hosts配置的主机组，同时我们也可以通过 -i 参数指定我们自己的hosts文件配置

**9.说说mvc架构原理（mtv）**

MVT

Django是一款python的web开发框架

与MVC有所不同，属于MVT框架

m表示model，负责与数据库交互

v表示view，是核心，负责接收请求、获取数据、返回结果

t表示template，负责呈现内容到浏览器

MVC

大部分开发语言中都有MVC框架

MVC框架的核心思想是：解耦降低各功能模块之间的耦合性，方便变更，更容易重构代码，最大程度上实现代码的重用

m表示model，主要用于对数据库层的封装

v表示view，用于向用户展示结果

c表示controller，是核心，用于处理请求、获取数据、返回结果

**10.你们的cmdb主要监控哪些项，zabbix监控哪些，为什么决定用zabbix（好处都有啥）**

 用户信息管理，记录测试，开发，运维人员的用户表

 业务信息线管理，需要记录业务的详情

 项目信息管理，指定此项目用属于哪条业务线，以及项目详情

 应用信息管理，指定此应用的开发人员，属于哪个项目，和代码地址，部署目录，部署集群，依赖的应用，软件等信息

 集群信息管理，指定集群属于哪个项目，以及集群的Level（开发，测试，生产）

 主机信息管理，包括云主机，物理机，主机属于哪个集群，运行着哪些软件，主机管理员，连接哪些网络设备，云主机的资源池，存储等相关信息

 主机信息变更管理，主机的一些信息变更，例如管理员，所属集群等信息更改，连接的网络变更等

 网络设备信息管理，主要记录网络设备的详细信息，及网络设备连接的上级设备

 IP信息管理，IP属于哪个主机，哪个网段, 是否被占用等

现有运维的工具有监控zabbix ，日志分析logstash+elasticsearch+kibana，持续集成ci jenkins，批量部署配置工具ansible。zabbix可以监控到一些应用层，网络层，硬件层的一些故障报警而去快速的处理，通过logstash+zabbix插件的配合使用，我们也可以对log来进行一些监控，比如某个log的关键字的过滤，error等错误的报警，或者http code状态码的报警，也能事先去发现一些问题，来快速的去处理响应。Zabbix现在监控了我们线上所有服务的运行状态，服务器的性能等，都能进行报警监控。  
 在规模较小，业务环境不复杂的情况下，可以以监控系统为主，但**规模较大，变化较快，必须以CMDB为主**。

zabbix**能同时提供Agent、SNMP、IPMI、JMX、WEB、SSH、TELNET、扩展监控脚本，并且能自定义的开源监控系统**

**能有一套即满足CMDB基础功能的，又可以与Zabbix，发布系统，业务系统相结合的CMDB，才是用户真正想要的CMDB系统**。

**9..ansible的协议是什么**

SSH 协议（或者 Kerberos、LDAP）

**11.说一下Jenkins你做了些什么？Jenkins上你们有几个job**

**12.Jenkins中如何实现一键部署的（脚本是什么），Jenkins的版本**

**13.如何提高python的运行效率**

使用生成器；关键代码使用外部功能包（Cython，pylnlne，pypy，pyrex）；针对循环的优化--尽量避免在循环中访问变量的属性

**13.描述数组、链表、队列、堆栈的区别？**

数组与链表：是数据存储方式的概念，数组在连续的空间中存储数据，而链表可以在非连续的空间中存储数据；

队列和堆栈是描述数据存取方式的概念，队列是先进先出，而堆栈是后进先出；队列和堆栈可以用数组来实现，也可以用链表实现。

**14.Python是如何进行内存管理的**

一、垃圾回收：

python不像C++，Java等语言一样，他们可以不用事先声明变量类型而直接对变量进行赋值。对Python语言来讲，对象的类型和内存都是在运行时确定的。这也是为什么我们称Python语言为动态类型的原因（这里我们把动态类型可以简单的归结为对变量内存地址的分配是在运行时自动判断变量类型并对变量进行赋值）。

二、引用计数：

Python采用了类似Windows内核对象一样的方式来对内存进行管理。每一个对象，都维护这一个对指向该对对象的引用的计数。当变量被绑定在一个对象上的时候，该变量的引用计数就是1，(还有另外一些情况也会导致变量引用计数的增加),系统会自动维护这些标签，并定时扫描，当某标签的引用计数变为0的时候，该对就会被回收。

三、内存池机制

Python的内存机制以金字塔行，-1，-2层主要有操作系统进行操作，第0层是C中的malloc，free等内存分配和释放函数进行操作；第1层和第2层是内存池，有Python的接口函数PyMem\_Malloc函数实现，当对象小于256K时有该层直接分配内存；第3层是最上层，也就是我们对Python对象的直接操作；在 C 中如果频繁的调用 malloc 与 free 时,是会产生性能问题的.再加上频繁的分配与释放小块的内存会产生内存碎片. Python 在这里主要干的工作有:如果请求分配的内存在1~256字节之间就使用自己的内存管理系统,否则直接使用 malloc.这里还是会调用 malloc 分配内存,但每次会分配一块大小为256k的大块内存.经由内存池登记的内存到最后还是会回收到内存池,并不会调用 C 的 free 释放掉.以便下次使用.对于简单的Python对象，例如数值、字符串，元组（tuple不允许被更改)采用的是复制的方式(深拷贝?)，也就是说当将另一个变量B赋值给变量A时，虽然A和B的内存空间仍然相同，但当A的值发生变化时，会重新给A分配空间，A和B的地址变得不再相同

**15.django 中当一个用户登录 A 应用服务器（进入登录状态），然后下次请求被 nginx 代理到 B 应用服务器会出现什么影响？**

如果用户在A应用服务器登陆的session数据没有共享到B应用服务器，那么之前的登录状态就没有了

**16.跨域请求问题django怎么解决的（原理）**

启用中间件

post请求

验证码

表单中添加{%csrf\_token%}标签

**17请解释或描述一下Django的架构**

对于Django框架遵循MVC设计，并且有一个专有名词：MVT

M全拼为Model，与MVC中的M功能相同，负责数据处理，内嵌了ORM框架

V全拼为View，与MVC中的C功能相同，接收HttpRequest，业务处理，返回HttpResponse

T全拼为Template，与MVC中的V功能相同，负责封装构造要返回的html，内嵌了模板引擎

**18.django对数据查询结果排序怎么做，降序怎么做，查询大于某个字段怎么做**

排序使用order\_by()

降序需要在排序字段名前加-

查询字段大于某个值：使用filter(字段名\_gt=值)

**19.说一下Django，MIDDLEWARES中间件的作用？**

中间件是介于request与response处理之间的一道处理过程，相对比较轻量级，并且在全局上改变django的输入与输出。

**20.你对Django的认识？**

Django是走大而全的方向，它最出名的是其全自动化的管理后台：只需要使用起ORM，做简单的对象定义，它就能自动生成数据库结构、以及全功能的管理后台。

Django内置的ORM跟框架内的其他模块耦合程度高。应用程序必须使用Django内置的ORM，否则就不能享受到框架内提供的种种基于其ORM的便利；理论上可以切换掉其ORM模块，但这就相当于要把装修完毕的房子拆除重新装修，倒不如一开始就去毛胚房做全新的装修。

Django的卖点是超高的开发效率，其性能扩展有限；采用Django的项目，在流量达到一定规模后，都需要对其进行重构，才能满足性能的要求。

Django适用的是中小型的网站，或者是作为大型网站快速实现产品雏形的工具。

Django模板的设计哲学是彻底的将代码、样式分离； Django从根本上杜绝在模板中进行编码、处理数据的可能。

**21.Django重定向你是如何实现的？用的什么状态码？**

使用HttpResponseRedirect

redirect和reverse

状态码：302,301

**22.ngnix的正向代理与反向代理？**

正向代理 是一个位于客户端和原始服务器(origin server)之间的服务器，为了从原始服务器取得内容，客户端向代理发送一个请求并指定目标(原始服务器)，然后代理向原始服务器转交请求并将获得的内容返回给客户端。客户端必须要进行一些特别的设置才能使用正向代理。

反向代理正好相反，对于客户端而言它就像是原始服务器，并且客户端不需要进行任何特别的设置。客户端向反向代理的命名空间中的内容发送普通请求，接着反向代理将判断向何处(原始服务器)转交请求，并将获得的内容返回给客户端，就像这些内容原本就是它自己的一样。

**23.Django 本身提供了 runserver，为什么不能用来部署？**

runserver 方法是调试 Django 时经常用到的运行方式，它使用 Django 自带的WSGI Server 运行，主要在测试和开发中使用，并且 runserver 开启的方式也是单进程 。

uWSGI 是一个 Web 服务器，它实现了 WSGI 协议、uwsgi、http 等协议。注意 uwsgi 一种通信协议，而 uWSGI 是实现 uwsgi 协议和 WSGI 协议的 Web 服务器。uWSGI 具有超快的性能、低内存占用和多 app 管理等优点，并且搭配着 Nginx就是一个生产环境了，能够将用户访问请求与应用 app 隔离开，实现真正的部署 。相比来讲，支持的并发量更高，方便管理多进程，发挥多核的优势，提升性能。