## 2018.6.1

#### 笔试题

1. 什么是多态

答:多态性是指具有不同功能的函数可以使用相同的函数名，这样就可以用一个函数名调用不同内容的函数。在面向对象方法中一般是这样表述多态性：向不同的对象发送同一条消息，不同的对象在接收时会产生不同的行为（即方法）。也就是说，每个对象可以用自己的方式去响应共同的消息。所谓消息，就是调用函数，不同的行为就是指不同的实现，即执行不同的函数

1. 写一段程序，判断字符串内左右括号是否配对

答:

class SStack():

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

self.\_\_elem = []

def is\_empty(self):

return self.\_\_elem == []

def top(self):

return self.\_\_elem[-1]

def push(self,elem):

self.\_\_elem.append(elem)

def pop(self):

return self.\_\_elem.pop()

def kuohao(text):

kinds = "()[]{}"#用来定义出现的括号，因为待匹配的字符中含有其他的字符，我们值检查括号是否匹配，而且是只有出现括号后再进行匹配

zuo = "([{" #定义左括号，如果是左括号就入栈

dict0 = {")":"(","]":"[","}":"{"}#匹配字典，这个字典定义了匹配规则，如果字典的键值对匹配就可以认定括号是匹配的

def pipei(text): #将等待匹配的文本输入，这个函数的目标是从文本中过滤出括号

i,text\_len = 0,len(text) #扫描指针用来记录匹配位置

while True:

while i< text\_len and text[i] not in kinds: #用来寻找到括号

i += 1

if i >= text\_len: #如果字符串中没有包含括号则结束

return

yield text[i],i #返回括号字符和字符对应的下标

i += 1

st = SStack()

for text0,i in pipei(text):#获取得到的符号进行匹配,因为pipei（）是一个含有yield函数，所以是一个生成器，调用它会产生一个可迭代的对象

if text0 in zuo: #如果是左括号就让它入栈

#print(text0)

st.push(text0)

elif st.pop() != dict0[text0]:#如果是右括号，就弹出栈顶元素进行匹配检查

print("本次不匹配")

return False #遇到不匹配的，就直接退出函数，结束匹配

print("所有的括号都已经匹配完毕，匹配成功！") #如果函数还能够执行到这里说明所有的括号都是匹配的

return True

#kuohao("({{[]}})")

kuohao("[{}]")

1. 写一段程序，求最大值，不可使用内置函数

答：def getMax(arr):

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

for j in range(i+1,len(arr)):

first=int(arr[i])

second=int(arr[j])

if first<second:

arr[i]=arr[j]

arr[j]=first

print (arr[0])

arr=[19,29,30,48]

getMax(arr)

1. 进程和线程的区别

答：

运行方式不同

进程不能单独执行，它只是资源的集合。

进程要操作CPU，必须要先创建一个线程。

所有在同一个进程里的线程，是同享同一块进程所占的内存空间。

关系

进程中第一个线程是主线程，主线程可以创建其他线程；其他线程也可以创建线程；线程之间是平等的。

进程有父进程和子进程，独立的内存空间，唯一的标识符：pid。

速度

启动线程比启动进程快。

运行线程和运行进程速度上是一样的，没有可比性。

线程共享内存空间，进程的内存是独立的。

创建

父进程生成子进程，相当于复制一份内存空间，进程之间不能直接访问

创建新线程很简单，创建新进程需要对父进程进行一次复制。

一个线程可以控制和操作同级线程里的其他线程，但是进程只能操作子进程。

交互

同一个进程里的线程之间可以直接访问。

两个进程想通信必须通过一个中间代理来实现。

1. 给定一个目录，找出目录中（包含子目录中）文件名以‘xxxx’开头的，并且文件名中包含今天日期的（形式为’yyyy-mm-dd’）文件名。

答：find \*-\*

#### 面试题

1. 介绍多线程

答：多线程就是指一个进程中同时有多个执行路径（线程）正在执行。

1. 用自己的话来讲述什么是面向对象编程

答：[面向对象](https://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E5%AF%B9%E8%B1%A1" \t "https://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E5%AF%B9%E8%B1%A1%E6%80%9D%E6%83%B3/_blank)的[程序设计语言](https://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F%E8%AE%BE%E8%AE%A1%E8%AF%AD%E8%A8%80" \t "https://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E5%AF%B9%E8%B1%A1%E6%80%9D%E6%83%B3/_blank)必须有[描述](https://baike.baidu.com/item/%E6%8F%8F%E8%BF%B0/8928757" \t "https://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E5%AF%B9%E8%B1%A1%E6%80%9D%E6%83%B3/_blank)对象及其相互之间关系的语言成分。这些程序设计语言可以归纳为以下几类：系统中一切事物皆为对象；对象是属性及其操作的封装体；对象可按其性质划分为类，对象成为类的实例；实例关系和继承关系是对象之间的静态关系；消息传递是对象之间动态联系的唯一形式，也是计算的唯一形式；方法是消息的序列。

1. 深拷贝与浅拷贝

答：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. Session 的存储位置，怎样跨服务器共享

1. 基于NFS的Session共享

　　NFS是Net FileSystem的简称，最早由Sun公司为解决Unix网络主机间的目录共享而研发。

这个方案实现最为简单，无需做过多的二次开发，仅需将共享目录服务器mount到各频道服务器的本地session目录即可，缺点是NFS依托 于复 杂的安全机制和文件系统，因此并发效率不高，尤其对于session这类高并发读写的小文件， 会由于共享目录服务器的io-wait过高，最终拖累前端WEB应用程序的执行效率。

　　基于数据库的Session共享

首选当然是大名鼎鼎的Mysql数据库，并且建议使用内存表Heap，提高session操作的读写效率。这个方案的实用性比较强，相信大家普 遍在 使用，它的缺点在于session的并发读写能力取决于Mysql数据库的性能，同时需要自己实现session淘汰逻辑，以便定时从数据表中更新、删除 session记录，当并发过高时容易出现表锁，虽然我们可以选择行级锁的表引擎，但不得不否认使用数据库存储Session还是有些杀鸡用牛刀的架势。

　 基于Cookie的Session共享

　　这个方案我们可能比较陌生，但它在大型网站中还是比较普遍被使用。原理是将全站用户的Session信息加密、序列化后以Cookie的方式， 统一 种植在根域名下（如：.host.com），利用浏览器访问该根域名下的所有二级域名站点时，会传递与之域名对应的所有Cookie内容的特性，从而实现 用户的Cookie化Session 在多服务间的共享访问。

　　这个方案的优点无需额外的服务器资源；缺点是由于受http协议头信心长度的限制，仅能够存储小部分的用户信息，同时Cookie化的 Session内容需要进行安全加解密（如：采用DES、RSA等进行明文加解密；再由MD5、SHA-1等算法进行防伪认证），另外它也会占用一定的带 宽资源，因为浏览器会在请求当前域名下任何资源时将本地Cookie附加在http头中传递到服务器。

　　基于Memcache的Session共享

　　Memcache由于是一款基于Libevent多路异步I/O技术的内存共享系统，简单的Key + Value数据存储模式使得代码逻辑小巧高效，因此在并发处理能力上占据了绝对优势，目前本人所经历的项目达到2000/秒 平均查询，并且服务器CPU消耗依然不到10%。

　　另外值得一提的是Memcache的内存hash表所特有的Expires数据过期淘汰机制，正好和Session的过期机制不谋而合，降低了 过期Session数据删除的代码复杂度，对比“基于数据库的存储方案”，仅这块逻辑就给数据表产生巨大的查询压力。

1. 数据库的索引原理是什么，什么表加索引，怎么判断索引是否提升查询速度。
2. http和https有什么不同

答：在URL前加https://前缀表明是用SSL加密的。 你的电脑与服务器之间收发的信息传输将更加安全。  
  
Web服务器启用SSL需要获得一个服务器证书并将该证书与要使用SSL的服务器绑定。  
http和https使用的是完全不同的连接方式,用的端口也不一样,前者是80,后者是443。http的连接很简单,是无状态的,...

HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议  
要比http协议安全

1. python2和python3的区别

答：1.性能

Py3.0运行 pystone benchmark的速度比Py2.5慢30%。Guido认为Py3.0有极大的优化空间，在字符串和整形操作上可   
以取得很好的优化结果。   
Py3.1性能比Py2.5慢15%，还有很大的提升空间。   
2.编码   
Py3.X源码文件默认使用utf-8编码，这就使得以下代码是合法的：   
    >>> 中国 = 'china'   
    >>>print(中国)   
    china   
3. 语法   
1）去除了<>，全部改用!=   
2）去除``，全部改用repr()   
3）关键词加入as 和with，还有True,False,None   
4）整型除法返回浮点数，要得到整型结果，请使用//   
5）加入nonlocal语句。使用noclocal x可以直接指派外围（非全局）变量

4、print函数

虽然print语法是Python 3中一个很小的改动，且应该已经广为人知，但依然值得提一下：Python 2中的print语句被Python 3中的print()函数取代，这意味着在Python 3中必须用括号将需要输出的对象括起来。

在Python 2中使用额外的括号也是可以的。但反过来在Python 3中想以Python2的形式不带括号调用print函数时，会触发SyntaxError。

1. Regex的优缺点

答：缺点

1.[正则表达式](https://www.baidu.com/s?wd=%E6%AD%A3%E5%88%99%E8%A1%A8%E8%BE%BE%E5%BC%8F&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)只适合匹配文本字面，不适合匹配文本意义：像匹配url，email这种纯文本的字符就很好，但比如匹配多少范围到多少范围的数字，如果你这个范围很复杂的话用正则就很麻烦。或者匹配html，这个是很多人经常遇到的，写一个复杂匹配html的正则很麻烦，不如使用针对特定意义的处理器来处理（比如写[语法分析器](https://www.baidu.com/s?wd=%E8%AF%AD%E6%B3%95%E5%88%86%E6%9E%90%E5%99%A8&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)，dom分析器等）  
  
2.容易引起性能问题：像.\*这种贪婪匹配符号很容易造成大量的回溯，性能有时候会有上百万倍的下降，编写好的[正则表达式](https://www.baidu.com/s?wd=%E6%AD%A3%E5%88%99%E8%A1%A8%E8%BE%BE%E5%BC%8F&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)要对正则引擎执行方式有很清楚的理解才可以  
  
3.正则的替换功能较差：甚至没有基本的截取字符串或者把首字母改变大小写的功能，这对于[url重写](https://www.baidu.com/s?wd=url%E9%87%8D%E5%86%99&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)引擎有时候是致命的影响  
  
优点：只要熟练应用[正则表达式](https://www.baidu.com/s?wd=%E6%AD%A3%E5%88%99%E8%A1%A8%E8%BE%BE%E5%BC%8F&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)，而且匹配的目标是纯文本，那么相比于写分析器来说，正则可以更快速的完成工作。还有在捕获字符串的能力，正则也可以很好的完成工作，比如截取url的域名或者其他的内容等等

1. 数据库的存储引擎都有哪些使用过那些。

InnoDB和MyISAM