**1．如何快速计算两个list的交集、并集**

简单的方法：

1 a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

2 b = [2, 5, 8, 11, 0]

3 # 交集（intersection）

4 intersection = [v for v in a if v in b]

5 # 并集（ union）

6 union = b.extend([v for v in a])

高效的方法：

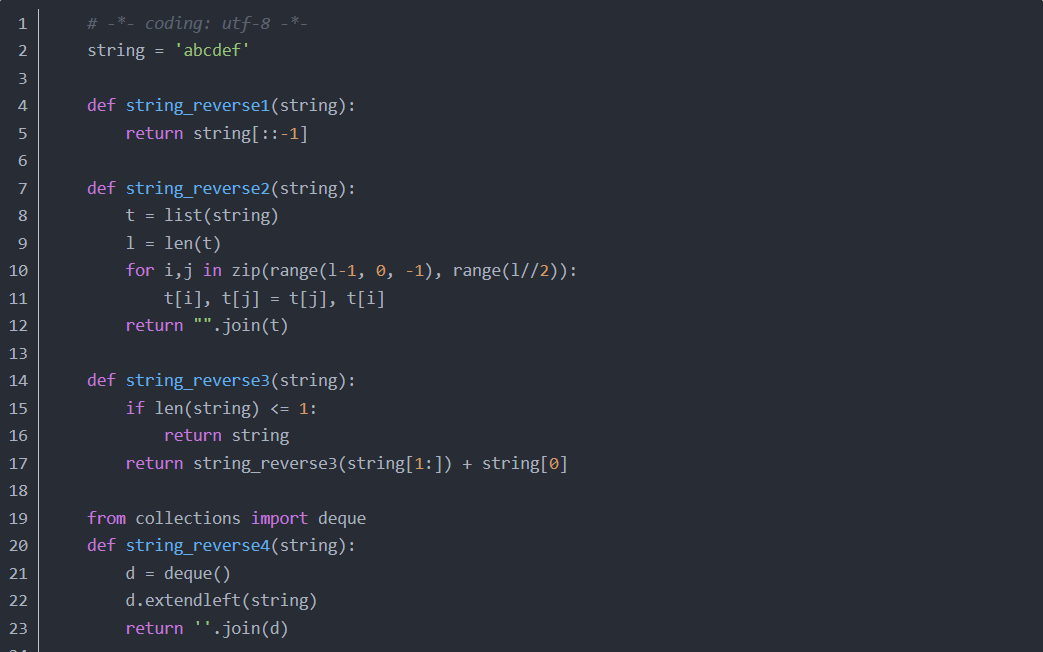
1 # 交集（intersection）

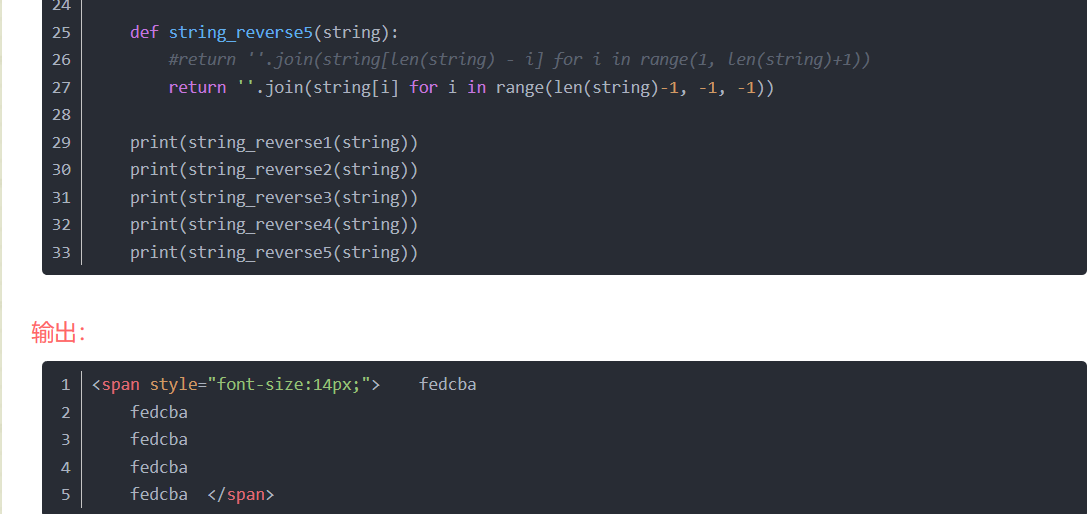
2 intersection = list(set(a).intersection(set(b)))

3 # 并集（union）

4 union = list(set(a).union(set(b)))

**2. 翻转一个字符串s = "abcdef"**





**3. 谈一谈你对list、tuple、set、dict的理解和区别**



**4. 举出几个Python自省的例子**

python自省是python具有的一种能力，使程序员面向对象的语言所写的程序在运行时,能够获得对象的类python型。Python是一种解释型语言。为程序员提供了极大的灵活性和控制力。

Python中常用的自省函数

1、help() 用来查看很多Python自带的帮助文档信息。

2、dir() 可以列出对象的所有属性。

3、type() 返回对象的类型。

4、id() 返回对象的“唯一序号”。对于引用对象来说,返回的是被引用对象的id()。

5、hasattr()和getattr() 分别判断对象是否有某个属性及获得某个属性值。

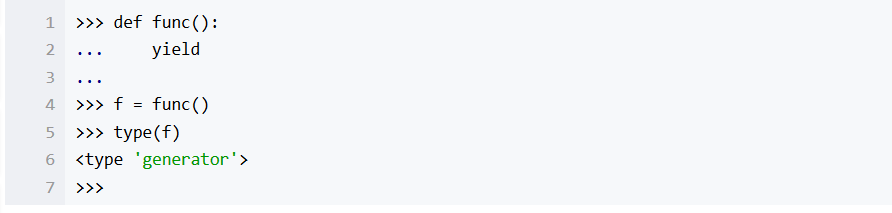
6、callable() 判断对象是否可以被调用。

7、isinstance() 可以确认某个变量是否有某种类型。

**5. 说说你对迭代器和生成器的理解**

迭代器是一个带状态的对象，它能在你调用next()方法的时候返回容器中的下一个值，任何实现了inter和next()（Python2中实现next()）方法的对象都是迭代器，iter返回迭代器自身，next返回容器的下一个值，如果容器中没有更多的元素了，则抛出Stopiteration异常，至于它们到底是如何实现的并不重要。正是因为他是通过next()来返回迭代器中的元素，所以它是一种延迟计算方式返回对象，这种特点对于大数据量元素进行遍历时具有明显优势，他不会一次性把所有元素载入内存，而是遍历一个载入一个，大大降低了内存的占用。

生成器，简单来说就是使用了yield关键字的函数，都叫做生成器。例如：



yield的作用是发起当前执行的函数，并返回，直到调用next()，继续执行后续指令，直到再次遇到yield或者抛出StopIteration异常。上面的例子第一次运行生成器f，需要首先调用f.next()启动生成器。

生成器的优点：代码实现更加简洁，可以提高代码的可读性。同时当然也具有迭代器的优点，大量数据遍历时内存占用少。

需要注意的是：生成器一定是迭代器，但是迭代器不一定是生成器，因为创建一个迭代器只需要实现**iter**和**next**()方法就可以了，并不一定要使用yield实现。生成器的唯一注意事项就是：生成器只能遍历一次。

**6. 实现一个二分查找**

二分查找：在一段数字内，找到中间值，判断要找的值和中间值大小的比较。

如果中间值大一些，则在中间值的左侧区域继续按照上述方式查找。

如果中间值小一些，则在中间值的右侧区域继续按照上述方式查 找。

直到找到我们希望的数字。

Python实现二分查找算法，代码如下：



**7. 实现一个装饰器，用于鉴权（已经有鉴权函数user\_auth(user\_id,token)）**



补充注释：内存中存在的pool也只有一个

**9. 用一条命令关掉所有的Python进程**

quit()   exit()  执行到此命令时，程序终止。

如果是程序陷入[死循环](https://www.baidu.com/s?wd=%E6%AD%BB%E5%BE%AA%E7%8E%AF&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YdrjwWPy7WmWP9nWb1mvNW0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHRknjb1n1mYn1m4Pjn1PHbsn0)，想强制结束，则按Ctrl + C

**10. 在用git提交中，如何避免提交pyc文件**

一般来说每个Git项目中都需要一个“.gitignore”文件，这个文件的作用就是告诉Git哪些文件不需要添加到版本管理中。实际项目中，很多文件都是不需要版本管理的，比如Python的.pyc文件和一些包含密码的配置文件等等。这个文件的内容是一些规则，Git会根据这些规则来判断是否将文件添加到版本控制中。

有两种方法可以实现过滤掉Git里不想上传的文件：

针对单一工程排除文件，这种方式会让这个工程的所有修改者在克隆代码的同时，也能克隆到过滤规则，而不用自己再写一份，这就能保证所有修改者应用的都是同一份规则，而不是张三自己有一套过滤规则，李四又使用另一套过滤规则，个人比较喜欢这个。配置步骤如下：

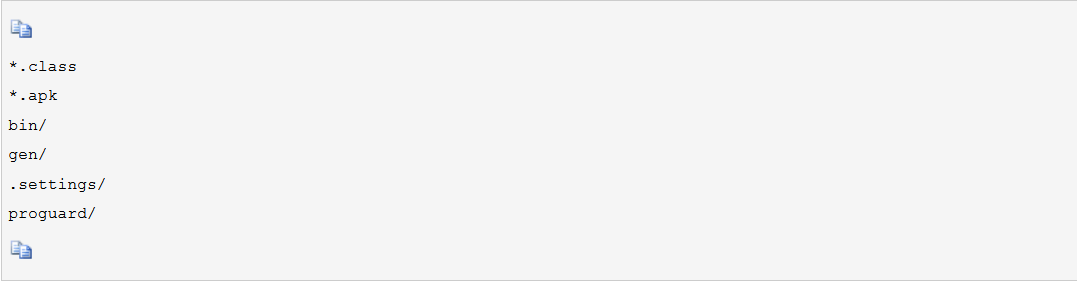
在工程根目录下建立.gitignore文件，将要排除的文件或目录 写到.gitignore这个文件中，其中有两种写入方法。

1. 使用命令行增加排除文件

排除以.class结尾的文件 echo “\*.class” >.gitignore (>> 是在文件尾增加,> 是删除已经存在的内容再增加)，之后会在当前目录下生成一个.gitignore的文件。

排除bin目录下的文件 echo “bin/” >.gitignore

b)最方便的办法是，用记事本打开，增加需要排除的文件或目录，一行增加一个，例如：



2）第二种方法

全局设置排除文件，这会在全局起作用，只要是Git管理的工程，在提交时都会自动排除不在控制范围内的文件或目录。这种方法对开发者来说，比较省事，只要一次全局配置，不用每次建立工程都要配置一遍过滤规则。但是这不保证其他的开发者在克隆你的代码后，他们那边的规则跟你的是一样的，这就带来了代码提交过程中的各种冲突问题。

配置步骤如下：

a）像方法（1）一样，也需要建立一个.gitignore文件，把要排除的文件写进去。

b）但在这里，我们不规定一定要把.gitnore文件放到某个工程下面，而是任何地方，比如我们这里放到了Git默认的Home路径下，比如：/home/wangshibo/hqsb\_ios

c）使用命令方式可以配置全局排除文件 git config --global core.excludesfile ~/.gitignore，你会发现在~/.gitconfig文件中会出现excludesfile = /home/wangshibo/hqsb\_ios/.gitignore。

说明Git把文件过滤规则应用到了Global的规则中。