Uuid(不唯一)模块和base64编码模块

uuid模块

概述：是128位的全局唯一标识符，通常由32字节的字母串表示，它可以保证时间和空间的唯一性，也称为 GUID

原理：通过MAC地址、时间戳、命名空间、随机数、伪随机数来保证生产的 ID的唯一性

作用：随机生成字符串，当成token使用，当成用户账号使用，当成订单号使用等（要求不相同字符串）

算法：

1、uuid1()基于时间戳
有MAC地址，当前时间错，随机数字，可以保证全球范围内的唯一性。但是由于 MAC地址的使用时带来安全问题，局域网中可以使用 IP来代替MAC
2、uuid2()基于分布式计算环境DCE
算法和uuid1相同，不同的是把时间戳的前四位换位 POSIX的UID，实际当中很少使用，注意， Python中没有这个函数
3、uuid3()基于名字和MD5散列值
通过计算名字和命名空间的MD5散列值得到的，保证了同一命名空间中不同名字的唯一性，和不同命名空间的唯一性，但同一命名空间的相同名字生成相同的 uuid
4、uuid4()基于随机数
有伪随机数得到的，有一定重复概率的，这个概率并且是可以计算出来的
5、uuid5()基于名字和SAH1散列值
算法和uuid3()相同，不同的是使用SHA1算法

使用经验：

1、python中没有基于DCE的，所以uuid2()可以忽略
2、uuid4()存在概率性重复，由于无映射性，最好不用
3、如果在全局的分布式环境下，最好使用 uuid1()
4、若名字的唯一性要求，最好使用 uuid3()或uuid5()

Floating Topic

base64模块

概述：用记事本打开图片等文件，看到一坨乱码，因为二进制文件包含很多无法显示的内容。所以想让记事本能处理二进制数据，就需要将二进制字符串转换。 base64是一种比较常见的二进制编码方法

编码原理：

编码原理：
一个包含64个字符的数组
['A', 'B', ……, 'a', 'b', ……, '0', '1', ……, '+', '/']

对二进制数据进行处理，每个三个字节一组，一组就是 3x8=24bit，划为4组，每组正好6bit

得到4个数字作为索引，然后查表，获得相应的 4个字符，就是编码后的字符串

作用：
适用于小段内容的编码，比如数字证书签名， cookie，网页中传输的少量二进制数据

注意：base64是一种通过查表的编码方法，不能用于加密，即使修改了字符对照表页不行。

注意：

```
#如果要编码的二进制不是3的倍数，怎么办？
#答：base用\x00字节在末尾补足，在编码的末尾加上 1 个或2 个等号表示补了多少个字节，解码时会自动去掉
# 由于标准base64编码后可能出现字符+和/，在URL中就不能直接作为参数
# 提供urlsafe_b64encode编码，保证url的安全，将+和/替换成-和_，提供urlsafe_b64decode进行url安全解码
```