1. 什么是僵尸进程？

如果子进程先结束而父进程后结束，即子进程结束后，父进程还在继续运行但是并未调用wait/waitpid那么子进程就会变成僵尸进程。

注意unix一般会在任何进程结束时都回收资源，但是仍然会保存pid等信息等待其父进程调用wait/waitpid来释放掉最后的资源。如果出现父进程早于子进程结束，那么子进程不会成为僵尸进程，它直接被init进程接管。

危害：

不释放pid等信息，进程号一直被占用，然而系统的进程号是有限的。

处理：

僵尸进程是无法被kill掉的，可以杀掉父进程来使用init接管僵尸进城。

1. python的GIL你是怎么理解的？

python多线程有一个GIL全局解释器锁，意思是任意时间只能有一个线程使用解释器。通常来讲GIL对python多线程无疑是一种抑制，没有单线程效率高，但是有一种特例就是多线程全部都是IO密集型时是有正面效果的，但只要存在至少一个CPU密集型线程，效率就会降低。

解决方案：

①multiprocess替代thread，但是多进程会出现通信附加消耗的问题，需要维护一个queue或者共享内存的方法。

②并行计算需求强的部分用c写。

1. python是如何进程内存管理的？

python中的内存管理机制有两套方案：

①内存池

Python引入了内存池机制mempry pool，即Pymalloc，用于管理对小块内存的申请和释放

当创建大量消耗小内存对象是，频繁调用new/malloc会导致大量的内存碎片，致使效率低下。内存池的概念就是预先在内存中申请一定数量的，大小相等的内存块留作备用，当有新内存申请时，就先从内存池中分配内存给这个需求，不够了久之后再申请新的内存。

②直接申请

内存池是针对小于256bits小对象的，如果大于256bits会直接调用new/malloc申请内存

同理关于释放内存：

释放内存时及一个对象的引用计数为0时，python就会调用它的析构函数，在析构时针对小对象如果是从内存池申请来的同样要放回内存池中，避免频繁的释放动作。

1. python2与python3的区别？

①性能

py3略逊py2一筹（怎么个慢还不清楚）

②编码

py3源码文件默认支持utf-8

③语法

去除<>，全部用!=

去除``，全部改用repr()

关键词加入as和with，还有True、False、None

整型除法返回浮点数，要得到整型结果使用//

加入nonlocal语句。使用nonlocal x可以直接指派外围（非全局）变量，而global则不是，global是指向全局变量的，一定要区别

去除print语句，加入print()函数

去除exec语句，改为exec()函数

改变顺序操作符行为，例如x<y，当x和y类型不匹配时抛出TypeError而不是返回 bool值

改变输入函数删除row\_input用input替代

去除元组参数解包。不能def(a, (b, c)):pass这样定义函数了

py3中八进制表示必须是0o...而py2中是0...，当然了既然改了表示那么oct()函数也相应改

新的super()可以不用传参数

支持class decorator用法跟函数decorator一样

④数据类型

去除long，都是int，可以把int看成py2中的long

新增bytes类型，str和bytes可以调用encode和decode相互转换

⑤异常

所有异常从BaseException继承并删除了StardardError

用 raise Exception(args)代替 raise Exception, args语法

捕获异常的语法改变，引入了as关键字来标识异常实例

⑥模块变动

移除了cPickle模块，可以使用pickle模块代替

.......

⑦其他

xrange() 改名为range()

zip()、map()和filter()都返回迭代器。而apply()、 callable()、coerce()、 execfile()、reduce()和reload ()函数都被去除了

废弃file类型