## 1. 什么是GIL？GIL对多线程的影响

GIL全称Global Interpreter Lock（全局解释器锁）。GIL和Python语言没有任何关系，只是因为历史原因导致在官方推荐的解释器Cpython中遗留的问题。每个线程在执行的过程中都需要先获取GIL，保证同一时刻只有一个线程可以执行代码，但是当遇到IO阻塞会自动的释放GIL锁，所以使用多线程还是比单线程的效率要高。如果想发挥多核CPU资源，可以使用多进程。为了避免受GIL的影响可以不用官方推荐的Cpython，或者用其他语言来实现。

计算密集型建议采用进程

IO密集型建议采用线程或者协程

## 2.深拷贝和浅拷贝

浅拷贝是对一个对象的顶层（外层）拷贝，只是拷贝了引用，并没有拷贝内容。

变量的赋值是地址的引用，也算是一种浅拷贝。

copy.copy()

深拷贝则是对一个对象深层（递归）的拷贝。

copy.deepcopy()

可变类型：列表、字典

不可变类型：数字类型、字符串型、元组

如果是可变类型，浅拷贝是指向，而深拷贝是完全拷贝

如果是**纯的**不可变类型，那么无论是浅拷贝还是深拷贝，都只是指向同一个地址。如果不可变类型里面还存在可变类型，则浅拷贝是指向，而深拷贝则为完全拷贝。

（了解）列表切片、字典的copy方法均属于浅拷贝

## 3.私有化

xx: 公有变量

\_x: 单前置下划线,私有化属性或方法，from 模块 import \*禁止导入，类对象和子类可以访问。

\_\_xx：双前置下划线，避免与子类中的属性命名冲突，无法在外部直接访问(名字重整所以访问不到，\_类名\_\_xx)

\_\_xx\_\_:双前后下划线，用户名字空间的魔法对象或属性。例如:\_\_init\_\_ , \_\_ 不要自己发明这样的名字。

xx\_:单后置下划线,用于避免与Python关键词的冲突，不推荐使用。

## 4.模块搜索顺序

import sys

sys.path # 返回查找模块的列表目录，列表中的路径的先后顺序代表了python解释器在搜索模块时的先后顺序。

sys.path.append('/home/itcast/xxx') # 在列表最后追加搜索目录

sys.path.insert(0, '/home/itcast/xxx') # 可以确保先搜索这个路径

## 5.重新导入模块

from imp import reload

reaload(模块名)

## 6.多模块开发时注意事项

通过from 模块 import 变量，此时相当于给一个变量赋值，如果在程序中修改了变量的值就导致这个变量成了局部变量，跟其他模块就不共享了。**如果多模块开发时想导入变量，建议使用import 模块名的方式，然后通过模块名.变量的方式去调用。**

## 7.面向对象的三大特性（封装、继承、多态）

封装就是把方法和属性封装到类的内部，只需要在类的外部，通过对象即可调用。继承实现了代码的重用。子类可以继承父类，并且可以继承多个父类即多继承，子类可以使用父类所拥有的属性和方法。多态是以继承和重写父类方法为前提，增加了代码的灵活度，只是一种调用技巧。