## Python 面试基础

1. **什么是PEP8？**

答：PEP8是一种编码规范，目的是为了让代码的可读性更高，更加规范。

1. **什么是pickling和unpickling？**

答：这是Python Pickle模块的功能。Pickle可以将几乎所有Python对象转换为字符串，让用户在编程时便于保存各种类型的数据对象。

pickling: Pickle 模块读入任何 Python 对象，将它们转换成字符串，然后使用 dump 函数将其转储到一个文件中。

unpickling: 从存储的字符串文件中提取原始 Python 对象的过程,叫做 unpickling 。

1. **Python是如何管理内存的？**

答：Python的内存是由私有heap空间管理的。所有的Python对象和数据结构都在一个私有heap中，程序员没有访问heap的权限，只有解释器才能进行操作。

为Python的heap空间分配内存是由Python内存管理模块进行的，其核心API会提供一些访问该模块的方法供程序员使用。

Python有自带的垃圾回收机制，它回收并释放没有被使用的内存，让他们能够被其他程序使用。

1. **Python中的命名空间是什么？**

答：命名空间是用于确保名称唯一以避免命名冲突的命名系统。

1. **什么是PYTHONPATH？**

答：这是一个环境变量，在导入模块时使用。每当导入模块时，都会查找PYTHONPATH以检查各个目录中是否存在导入的模块。一般情况下，解释器会用它来确定要加载哪个模块。

1. **什么是lambda函数？**

答：lambda被称为匿名函数。此函数可以具有任意数量的参数，但是只能有一个表达式，也只能返回单个表达式值。

使用lambda函数的好处有：(1). lambda函数比较轻便，即用即扔，很适合需要完成某一项简单功能；(2). lambda是匿名函数，一般用来给filter，map，reduce这样的函数式编程服务；(3). 作为回调函数，可以传递给某些应用，比如消息处理等。

1. **Python中的self是什么？**

答：self是类的实例或对象。在Python中，self包含在第一个参数中。但是，Java中的情况并非如此，它是可选的。它有助于区分具有局部变量的类的方法和属性。init方法中的self变量引用新创建的对象，而在其他方法中，它引用其方法被调用的对象。

1. **Python中的列表和元组有什么区别？**

答：列表（list）是可变的，即可以编辑的，而元组（tuple）是不可变的。

1. **Python数组和列表有什么区别？**

答：在Python中，数组和列表具有相同的数据存储方式。但是，数组只能容纳一个数据类型元素，而列表可以容纳任何数据类型元素。

1. **为什么使用\* args，\*\* kwargs？**

答：当我们不确定将多少个参数传递给函数，或者我们想要将存储的列表或参数元组传递给函数时，我们使用\* args。\*\*当我们不知道将多少关键字参数传递给函数时使用kwargs，或者它可以用于将字典的值作为关键字参数传递。标识符args和kwargs是一个约定，其实也可以使用\* bob和\*\* billy。

1. **什么是负指数，功能是什么？**

答：Python中的序列是索引的，它由正数和负数组成。积极的数字使用'0'作为第一个索引，'1'作为第二个索引，进程继续使用。

负数的索引从'-1'开始，表示序列中的最后一个索引，' - 2'作为倒数第二个索引，序列像正数一样前进。

负索引用于从字符串中删除任何换行符，并允许该字符串除了作为S [： - 1]给出的最后一个字符。负索引还用于显示索引以正确的顺序表示字符串。

1. **NumPy中有哪些操作Python列表的函数？**

答：Python的列表是高效的通用容器。它们支持（相当）有效的插入，删除，追加和连接，Python的列表推导使它们易于构造和操作。

它们有一定的局限性：它们不支持像素化加法和乘法等“向量化”操作，并且它们可以包含不同类型的对象这一事实意味着Python必须存储每个元素的类型信息，并且必须执行类型调度代码在对每个元素进行操作时。

NumPy不仅效率更高; 它也更方便。你可以免费获得大量的向量和矩阵运算，这有时可以避免不必要的工作。它们也得到有效实施。

NumPy数组更快，你可以使用NumPy，FFT，卷积，快速搜索，基本统计，线性代数，直方图等内置。

1. **如何在Python中实现多线程？**

答：Python有一个名为Global Interpreter Lock（GIL）的结构。GIL确保每次只能执行一个“线程”。一个线程获取GIL执行相关操作，然后将GIL传递到下一个线程。

虽然看起来程序被多线程并行执行，但它们实际上只是轮流使用相同的CPU核心。

所有这些GIL传递都增加了执行的开销。这意味着多线程并不能让程序运行的更快。

1. **Python 3.X 中的默认编码是什么？**

答：Python 3.X的解释器是用 UTF-8 编码的。

### 解释一下Python中map和reduce？

答：map()是Python内置的高阶函数，它接收一个函数f和一个list，并通过把函数f依次作用在list的每个元素上，得到一个新的list并返回。（Python 3中返回的迭代器）

reduce()函数接收的参数和 map()类似，一个函数f和一个list，但行为和 map()不同， reduce()传入的函数f必须接收两个参数， reduce() 对list的每个元素反复调用函数f，并返回最终结果的值。python3使用 reduce需要导入，在 functools中。