1、描述一下python语言

Python是一种编程语言，它有对象、模块、线程、异常处理和自动内存管理。

它简洁、简单、方便、容易扩展，有许多自带的数据结构，而且它开源。

2、编程：写出你所了解的排序算法，并且使用python实现一种

排序的方法有：插入排序（直接插入排序、希尔排序），交换排序（冒泡排序、快速排序），选择排序（直接选择排序、堆排序），归并排序，分配排序（箱排序、基数排序）

直接插入排序：

def insert\_sort(array):

for i in range(len(array)):

for j in range(i):

if array[i] < array[j]:

array.insert(j, array.pop(i))

break

return array

array = [1,4,8,2]

print(insert\_sort(array))

冒泡排序：

def bubble\_sort(array):

for i in range(len(array)):

for j in range(i, len(array)):

if array[i] > array[j]:

array[i], array[j] = array[j], array[i]

return array

array = [1,4,8,2]

print(bubble\_sort(array))

3、编程：给一个字符串，编程实现它的所有字符组合可能

Ex：‘123’：组合就有1,2,3,12,13,23,123， 12和21是一样的，这里不考虑顺序

>>>import itertools

>>>list(itertools.combinations([1,2,3],1))

>>>list(itertools.combinations([1,2,3],2))

>>>list(itertools.combinations([1,2,3],3))

4、解释一下MVC开发模式

MVC的具体含义是：model+view+control，即模型+视图+控制

它们各自处理自己的任务：

（1）模型：模型持有所有的数据、状态和程序逻辑。模型独立于视图和控制器。

（2）视图：用来呈现模型。视图通常直接从模型中取得它需要显示的状态与数据。对于相同的信息可以有多个不同的显示形式或视图。

（3）控制器：位于视图和模型中间，负责接受用户的输入，将输入进行解析并反馈给模型，通常一个视图具有一个控制器。

MVC模式将它们分离以提高系统的灵活性和复用性，不使用MVC模式，用户界面设计往往将这些对象混在一起。MVC模式实现了模型和视图的分离，这带来了几个好处。

（1）一个模型提供不同的多个视图表现形式，也能够为一个模型创建新的视图而无须重写模型。一旦模型的数据发生变化，模型将通知有关的视图，每个视图相应地刷新自己。

（2）模型可复用。因为模型是独立于视图的，所以可以把一个模型独立地移植到新的平台工作。

（3）提高开发效率。在开发界面显示部分时，你仅仅需要考虑的是如何布局一个好的用户界面；开发模型时，你仅仅要考虑的是业务逻辑和数据维护，这样能使开发者专注于某一方面的开发，提高开发效率。

直接范范的说MVC适合不适合某个项目都是不准确的，所以我们在使用MVC模式的时候，要发挥其优点和长处！

5、描述一下你擅长的开发模式

所谓设计模式，就是一套被反复使用的代码设计经验的总结（情境中一个问题经过证实的一个解决方案）。使用设计模式是为了可重用代码、让代码更容易被他人理解、保证代码可靠性。设计模式使人们可以更加简单方便的复用成功的设计和体系结构。将已证实的技术表述成设计模式也会使新系统开发者更加容易理解其设计思路

- 工厂模式：工厂类可以根据条件生成不同的子类实例，这些子类有一个公共的抽象父类并且实现了相同的方法，但是这些方法针对不同的数据进行了不同的操作（多态方法）。当得到子类的实例后，开发人员可以调用基类中的方法而不必考虑到底返回的是哪一个子类的实例。

- 代理模式：给一个对象提供一个代理对象，并由代理对象控制原对象的引用。实际开发中，按照使用目的的不同，代理可以分为：远程代理、虚拟代理、保护代理、Cache代理、防火墙代理、同步化代理、智能引用代理。

- 适配器模式：把一个类的接口变换成客户端所期待的另一种接口，从而使原本因接口不匹配而无法在一起使用的类能够一起工作。

- 模板方法模式：提供一个抽象类，将部分逻辑以具体方法或构造器的形式实现，然后声明一些抽象方法来迫使子类实现剩余的逻辑。不同的子类可以以不同的方式实现这些抽象方法（多态实现），从而实现不同的业务逻辑。

6、说一下 left join 和 right join的区别

两者是数据库连接查询里的外连接，其中左连接是以左表为主显示查询结果left join，

用法格式为：select 字段名列表 from 表1 left join 表2 on 条件..

右连接是以右表为主显示查询结果right join，

用法格式为：select 字段名列表 from 表1 right join 表2 on 条件..

7、编程：从数据库中获取数据，存入‘a.txt’中

方式1. select user,host（数据信息） from mysql.user（表名）

into outfile "/var/lib/mysql-files/a.txt"（文件路径）

fields terminated by " : "

lines terminated by "\n";

方式2.

#coding=utf-8

import MySQLdb as db

import sys

import codecs

reload(sys)

sys.setdefaultencoding('utf-8')

def connectionDB():

conn=db.connect(host='localhost',user='root',db="test",passwd=123456,port=3306,charset="utf8")

cur = conn.cursor()

#开始的时候，在e:\get\_key\中生成一个 农学\_兽医学的txt文件

defau\_subject\_first = '医学'

defau\_subject\_second = '临床医学'

path = 'E:\\get\_key2\\' + defau\_subject\_first + '\_' + defau\_subject\_second + '.txt'

#注意，这里路径是e:\\get\_key2\\ 主要是\是一个特殊符号，我们需要转义

paper\_keywords1 = codecs.open(path.decode('utf-8'), 'wb',encoding="utf-8")

#在打开文件的时候，为了防止出现中文乱码的问题，用codecs.open去打开文件

#把结果先放到sql里面

sql = "SELECT name\_chinese,subject\_first ,subject\_second FROM periodical"

cur.execute(sql)

results = cur.fetchall()

#把结果写到txt里面

for result in results:

paper\_keywords1.write(result[0])

8、什么是线程？（尽可能的从多方面解释）

线程（Thread），有时被称为轻量级进程(Lightweight Process，LWP），是程序执行流的最小单元。一个标准的线程由线程ID，当前指令指针(PC），寄存器集合和堆栈组成。另外，线程是进程中的一个实体，是被系统独立调度和分派的基本单位，线程自己不拥有系统资源，只拥有一点儿在运行中必不可少的资源，但它可与同属一个进程的其它线程共享进程所拥有的全部资源。一个线程可以创建和撤消另一个线程，同一进程中的多个线程之间可以并发执行。由于线程之间的相互制约，致使线程在运行中呈现出间断性。线程也有就绪、阻塞和运行三种基本状态。就绪状态是指线程具备运行的所有条件，逻辑上可以运行，在等待处理机；运行状态是指线程占有处理机正在运行；阻塞状态是指线程在等待一个事件（如某个信号量），逻辑上不可执行。每一个程序都至少有一个线程，若程序只有一个线程，那就是程序本身。

线程是程序中一个单一的顺序控制流程。进程内有一个相对独立的、可调度的执行单元，是系统独立调度和分派CPU的基本单位指令运行时的程序的调度单位。在单个程序中同时运行多个线程完成不同的工作，也称为多线程