coding test：

coding test目标

请说明项目打包和执行的方式

请说明发布的方法

不限语言和平台

如有问题，请随时电话沟通

即使你的项目无法打包执行，请仍然提交您的代码

请确保您的github是公共可访问的

一、问题：

F(n) = F(n-1) + F(n-2), where F1 =1, F2 = 2

F1 =1

F2 =1

F3 =2

F4 =3

F5 =5

F6 =8

F7 =13

F8 =21

F9 =34

F10=55

F11=89

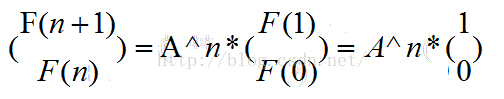
F12=144

可见index=12的F12是第一个包含3个数字的Fn，那么第一个包含1000个数字的Fn的index是多少？

答案:是4782， 代码见github, 在本地执行时耗时接近2分钟。

总体算法思路：

1. 将斐波拉契计算公式转换为矩阵的N-1次方



1. 使用快速幂算法，使用二进制比特移位操作，累乘基数，只对bit位为1进行乘法运算
2. 使用十进制字符串实现大数加减法
3. 使用karatsuba算法实现大数相乘(矩阵运算需要用到)
4. 通过矩阵N次方快速找到(start,end)这两个值，使得F(start)的位数小于数字长度，F(end)大于等于数字长度。然后，使用logN对数折半查找法找中间值，不断逼近n值。

二、将1，2，3，4，……，99，100的顺序排列，排序成100，1，99，2，98，3，……，51，50这样的非等差数列。要求:空间复杂度o（1），时间复杂度o（n）。

三、素数是自然数中大于1，且只能被正整数中的1和自身整除的自然数。

比如10以下的素数和为 ：2 + 3 + 5 + 7 = 17

请计算2,000,000以下的所有素数和

四、问答：

排序算法的平均时间复杂度的下限是多少？请尝试证明您的观点

答：平均时间复杂度下限为O(nlogn)

证明：

给定n个待排序元素，可能的排列结果有n!，经过第一次比较，可以确定两个

元素的排列顺序，剩余可能的排列结果又n!/2, 根据归纳法依次类推，经过m次

比较后，剩余可能为n!/(2^m)，直到n!/(2^m)<=1时，排序结果只剩余一种。比较

次数至少为Log(n!), 根据斯特林公式，当N足够大时，Log(n!)=O(nlogn), 最优情况

下，复杂度是o(nlogn)。

五、目前有线索池系统的需求，需要记录线索来源和线索信息，线索信息的内容除了客户名和多种（至少）一种联系方式外，并不明确信息的数量和内容。另有下单系统，系统中有客户名和一种联系方式，包含供应商信息等确定的60个不同类型字段。线索池的客户名和下单的客户名不一定相同。此外还有代理商网咯系统，会得到线索池分发的信息。

请尝试设计线索池系统，其中下单系统和代理商网络是旧有系统，有一定改造难度。总体目标是实现线索有效性的评价，尽可能提高线索的利用率（供应商对线索的有效利用）。

1. 线索池系统信息数量和内容不明确，考虑采用NoSQL数据库作为存储，使用JSON方式存储数据，即以后有新增字段不受影响，联系方式多种的话，则可以作为一个JSON的节点。
2. 线索池系统需要考虑如何与下单系统、代理商网络系统交换信息，首先线索池系统需要提供适合这两个系统的接口，根据具体情况和人员选择web api,web url页面，web service,数据库等不同接口供两个系统主动调用，提供基于客户名、联系方式等为条件的查询接口。
3. 如果需要线索池系统主动通知下单和代理商网络系统关于客户的线索信息，可以通过MQ的方式来单独建个模块，监听线索池系统的信息的新增，更新等消息，然后去主动调用另外两个系统的接口。这包括分发信息给代理商网络系统，如果代理商网络系统有多个部署，又没有统一的broker,则建议在代理商网络系统开发部署一个broker,通过MQ订阅，封装处理如何分发给不同的部署节点。
4. 考虑到下单系统和代理商网络系统旧系统，不需要对现有功能做大改造，可以考虑在订单系统或代理商网络系统增加关联表，设计订单与线索信息的关联(1对多), 在页面上增加URL链接或嵌入，直接根据客户信息，联系方式等关键字去调用线索池系统的接口或页面。

(5)如果线索池系统需要实时更新线索信息与订单的关联，可以考虑在订单系统数据库里增加触发器来生成消息事件，通过单独的worker去消费这个事件，来同步更新线索池系统数据库，确保线索与订单的信息有关联，可作为统计线索有效性的关键字段。

1. 请简述jvm模型，包括并发模型，请至少说明两项不涉及garbage的内容

Java代码是运行在Java虚拟机(JVM)上的，Java虚拟机通过解释执行（解释器）或编译执行（编译器）来完成。Java内存模型分为5个部分：方法区（Method Area），Java堆（Heap），Java栈（VM Stack），本地方法栈（Native Method Stack），程序计数器（PC 寄存器）

方法区（Method Area）：方法区是各个线程共享的区域，存放类信息，常量，静态常量，编译器编译后的代码等信息。

Java堆（Heap）：Java堆也是线程共享区域，类的实例存放在这里，一个系统会产生很多Java实例，因此Java堆的空间是最大的，如果Java堆的空间不足，就会抛出OutOfMemoryError异常。

线程私有区：

Java栈（VM Stack）：线程私有区域，生命周期与线程相同，一个线程对应一个Java栈，每执行一个方法就会向栈里压一个元素，这个元素叫“栈帧”，栈帧中包含了方法中保存了该方法调用的参数、局部变量和返回地址等信息，如果栈空间不足了就会抛出StackOverflowError异常。

本地方法栈（Native Method Stack）：和Java栈类似，本地方法栈是用来执行本地方法的，存放的方法调用本地方法接口，最终调用本地方法库，实现与操作系统，硬件交互的目的。

程序计数器：这里对应的类以及加载，实例对象，方法，静态变量去了该去的地方，那么问题来了，程序该怎么执行，哪个方法先执行，哪个方法后执行，这些指令执行的顺序就是PC寄存器在管，它的作用就是控制程序指令的执行顺序。

Java通过线程来实现并发，线程具有以下特性：

原子性：

不可切割。lock、unlock。。。等操作；synchronized块。

可见性：

一个线程修改了共享变量后，另外一个线程立即看得到。volatile/final/synchronized。

有序性：

本线程类观察，所有操作有序；其他线程观察就是无序。（线程内表现为有序）