# 实验二 多线程程序实验

## 实验内容

1. 用线程生成Fibonacci数列

要求：

用pthread线程库，按照第四章习题4.11的要求生成并输出Fibonacci数列。

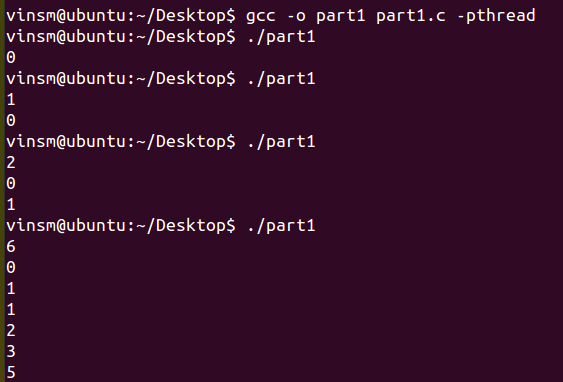
分析：

斐波那契数列为0，1，1，2，3，5……，可表示为f（0）=0，f（1）=1，f（n）=f（n-1）+f（n-2）。该程序从命令行输入中接收产生斐波那契数列的项数，然后创建一个新的线程来调用一个处理函数，进而生成斐波那契数列，放到一个全局的数组中。子线程执行完成后，父线程输出全局数组中的斐波那契数列。

使用pthread\_create()创建线程，给调用函数传的参数都是(void \*)&类型的，在函数中要进行转换；使用pthread\_join()阻塞调用线程，直到threadid所指定的线程终止；在处理函数中使用pthread\_exit(0)终止调用线程，而不是用return。

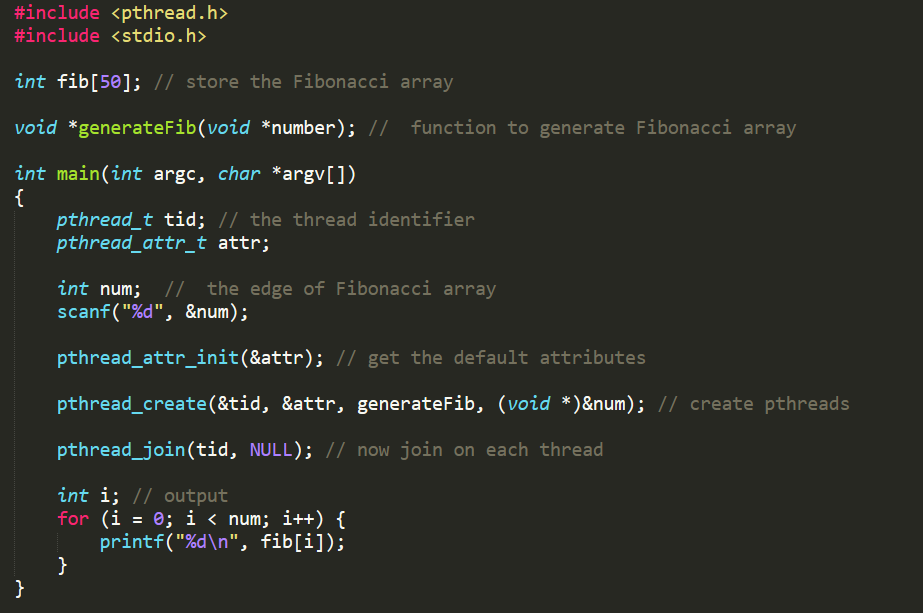
pthread\_t tid获得线程标识，pthread\_attr\_init(&attr)获得缺省的线程属性。

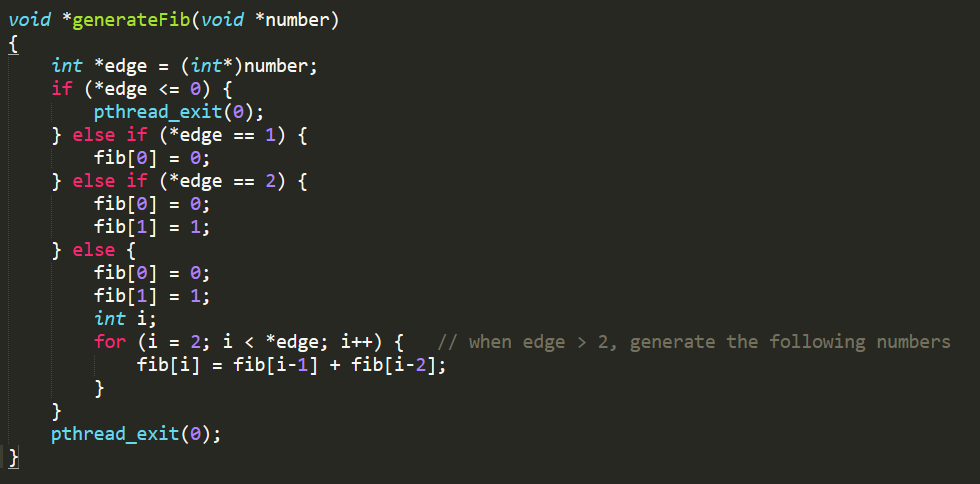
实验结果：



输入小于或等于0的数是非法的，不输出任何数列；输入1、2时，输出f（0）、f（0）和f（1）；输入大于2的数时，输出生成的斐波那契数列。

程序代码：





1. 多线程矩阵乘法

要求：

给定两个矩阵A和B，其中A是具有M行、K列的矩阵， B为K行、N列的矩阵， A和B的矩阵积为矩阵C， C为M行、N列。矩阵C中第i行、第j列的元素Cij就是矩阵A第i行每个元素和矩阵B第j列每个元素乘积的和，即



每个Cij的计算用一个独立的工作线程，因此它将会涉及生成M×N个工作线程。主线程(或称为父线程)将初始化矩阵A和B，并分配足够的内存给矩阵C，它将容纳矩阵A和B的积。这些矩阵将声明为全局数据，以使每个工作线程都能访问矩阵A、B和C。

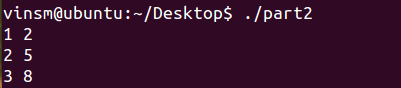
分析：

在父线程中，初始化两个用于相乘的矩阵，A是大小为3 \* 2的矩阵，B是大小为2 \* 2的矩阵，因此乘积矩阵result的大小为3 \* 2，它们都是全局的二维数组。

乘积矩阵每一个元素的计算用一个独立的工作线程，所以程序产生6个工作线程，每当需要计算乘积矩阵的一个元素时就产生一个线程并调用它。这里每生成一个线程就直接调用的原因是，当我把所要用的线程全部生成后再一个个调用的话，就会出现函数内部计算错误，而其他地方都没有出错，猜想是线程之间互相竞争导致的，所以要一个个线程生成并调用。

用一个结构体封装要传进函数的2个参数，传指针进去，函数内部用公式计算乘积矩阵的第[i, j]个元素。

实验结果：



这里矩阵A是 ， B是 ， 都是已经初始化好的。

程序代码：

