**哲学家就餐问题(Dining Philosophers)**

1. **问题重述**

五个哲学家围坐在一张圆桌周围，每个哲学家面前都有一碟通心粉。由于面条很滑，所以要两把叉子才能夹住。相邻两个碟子之间有一把叉子。哲学家的生活包括两种活动：即吃饭和思考。当一个哲学家觉得饿时，他就试图分两次去取他左边和右边的叉子，每次拿一把，但不分次序。如果成功地获得了两把叉子，他就开始吃饭，吃完以后放下叉子继续思考。

每个哲学家的行为定义为以下函数：void philosopher(int i)。请在多线程环境下实现描述以上行为的程序。

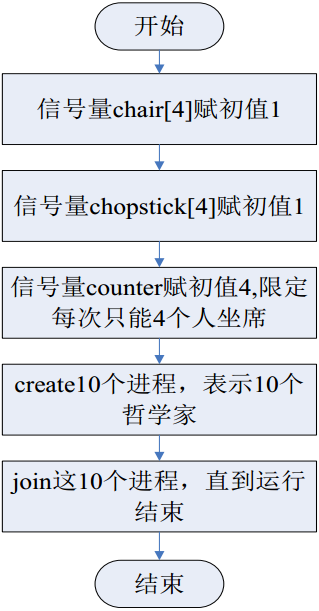
1. **项目要求及分析**

假如所有的哲学家都同时拿起左侧叉子，看到右侧叉子不可用，又都放下左侧叉子，等一会儿，又同时拿起左侧叉子，如此这般，永远重复。对于这种情况，即所有的程序都在无限期地运行，但是都无法取得任何进展，即出现饥饿，所有哲学家都吃不上饭。规定在拿到左侧的叉子后，先检查右面的叉子是否可用。如果不可用，则先放下左侧叉子，等一段时间再重复整个过程。

分析：当出现以下情形，在某一个瞬间，所有的哲学家都同时启动这个算法，拿起左侧的叉子，而看到右侧叉子不可用，又都放下左侧叉子，等一会儿，又同时拿起左侧叉子……如此这样永远重复下去。对于这种情况，所有的程序都在运行，但却无法取得进展，即出现饥饿，所有的哲学家都吃不上饭。

解决死锁问题:为了避免死锁，把哲学家分为三种状态，思考，饥饿，进食，并且一次拿到两只叉子，否则不拿。仅当一个哲学家左右两边叉子都可用时，才允许他拿叉子。这样要么一次占有两只叉子在吃面，然后释放所有资源；要么不占用资源。这样就不会导致死锁了。

1. **流程图**



1. **调试运行结果**

