# 第一题设计思路

1. 主要思路：多个临界资源，要么全部分配，要么一个都不分配，因此不会出现死锁的情形。
2. 即： 能同时拿到左右手的叉子才可以拿到叉子，不然拿不到叉子，即不会占用资源，从而避免死锁的发生。
3. 缺点： 如果1,3号轮流拿到叉子，有可能会导致2号一直都拿不到叉子。从而出现无限等待。

# 如何运行第一题

1. 在根目录下：
2. 执行：runghc.exe Philosopher.hs

# 第二题设计思路

1. 扫描输入的字符串

2. 获取优先级最高的“()”的内容，并将结果计算出来，

2.1 计算结果，先替换"n--m" 成 "n+m”，替换"n-m"成 "n+-m"

2.2 再按“\*+/”符号去切割字符串，没有“-”是因为2.1处理了。

2.3 解析替换后的结果，然后根据优先级"\*/+"，从左往右，计算字符串的结果

2.3 返回计算后的结果

3. 把计算出来的结果，替换掉2里面的"()"。

4. 如果经过3之后还有“()”,则再进行第二步

5. 没有"()"之后，就直接计算结果。

# 如何运行第二题

1. 进入到“threepenny-gui/samples“目录下
2. 执行：runhaskell Calculator.hs