HW4

PB2111686_赵卓

1.

- LINE C的输出为5。父进程fork()之后产生子进程，子进程复制父进程的所有资源并执行，子进程的pid为0。因此LINE C在子进程中执行，此时子进程创建了一个线程，该线程将value赋值为5，在LINE C中将value打印输出，因此输出为5。
- LINE P的输出为0。父进程的pid为子进程的pid值，因此非0，LINE P会在父进程中执行。而父进程的value和子进程的value是分开的，因此虽然子进程将value赋值为5，但是并不影响父进程的value。因此LINE P打印父进程的value为0。

2.

- 可能会导致饥饿现象。例如，当有多个reader和一个writer时，如果reader先执行，那么直到最后一个reader执行完毕时，readercount才会为0，reader才会发出signal(wrt)使writer执行。如果reader很多，writer会一直停留在wait(wrt)上无法执行导致饥饿。对于reader同理。

3.

- 对于图一，没有处于死锁状态，可能的两种执行顺序：
  - T2先执行完毕释放R2，T3获取R2执行完毕，然后释放R2，然后T1获取R2并执行完毕。因此顺序为T2，T3，T1。
  - T2先执行完毕释放R2，T1获取R2执行完毕，然后释放R2，然后T3获取R2并执行完毕。因此顺序为T2，T1，T3。
- 对于图二，没有处于死锁状态，可能的三种执行顺序：
  - T4执行完毕释放一份R2，T2获取R2执行完毕再释放，然后T1获取R2执行完毕，释放R1，然后T3获取R1执行完毕。因此顺序为T4，T2，T1，T3。
  - T4执行完毕释放一份R2，T1获取R2执行完毕再释放R2和R1，然后T2获取R2，T3获取R1，但是T2先执行完毕。因此顺序为T4，T1，T2，T3。
  - T4执行完毕释放一份R2，T1获取R2执行完毕再释放R2和R1，然后T2获取R2，T3获取R1，但是T3先执行完毕。因此顺序为T4，T1，T3，T2。

4.

- 对于Request(3,3,0)，不能允许，因为Request[0]>Available[0]，此时无法分配。
- 对于Request(0,2,0)，不能允许，因为给P0分配之后，Available变为[2,1,0]，由银行家算法检测安全性，第一步就无法找到Finish为true的进程，因此该分配不安全，不允许分配。