第二章

1. 为什么只有当信号量初始化为1时，才能实现互斥锁的功能（从wait和signal语句的定义来解释）。

答： 对于信号量S, wait的定义是： wait(S): while S≤0 do 循环测试

否则：S∶=S-1

Signal的定义是：signal(S)：S=S+1

当S初始化为1，第一个捕捉到S的进程执行wait(S) 操作，S=0。然后继续往下执行。

第二个进程执行wait(S)，根据定义，进入了循环测试，直到第一个进程释放S, 即signal(S)之后，第二个进程继续向下执行。

S把第一个程序中wait(S)和signal(S)之间的代码作为临界区给锁住了。

1. 在分时操作系统中，简要阐述一下多个用户如何共享一台计算机？（从并发和互斥的角度考虑，总共不要超过三句话）

答：在分时操作系统中，多个用户共享一台计算机。

**并发性**：同一时间间隔内，系统分成多个时间片，

每个时间片处理一个用户的响应。

**互斥性**：当某一用户的时间片用完后，必须暂停等待，此时系统处理下一个用户作业。

1. 写出进程三个基本状态之间的转换及其原因。

见教科书

4、对于进程管理，以下说法错误的是 （ D ）

A. 进程控制块PCB被同一进程的不同线程所共享。

B. 进程间的切换发生在内核态。

C. 子进程和父进程共享同一个进程的数据段和 代码段。

D. 在P、V操作中，同一个信号量的wait和signal两个操作，必须由同一个进程来完成。

原因：同一个信号S的捕捉和释放，虽然要求成对出现，但是可以由不同的进程来完成。详见书上的例题。

第三章

1．有五个进程A、B、C、D、E，到达时间和服务时间如下表所示，请按照**非抢占式的动态**高优先权优先算法，完成下表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 进程名称 | A | B | C | D | E |
| 进程情况 | 到达时间 | 0 | 1 | 2 | 3 | 4 |
| 服务时间 | 3 | 2 | 3 | 5 | 4 |
| 高优先权优先算法 | 等待时间 | 0 | 2 | 3 | 5 | 9 |
| 开始时间 | 0 | 3 | 5 | 8 | 13 |
| 完成时间 | 3 | 5 | 8 | 13 | 17 |
| 周转时间 | 3 | 4 | 6 | 10 | 13 |

当t=3时，A完成，B, C, D 已到达

W\_B=1+2/2=2;

W\_C=1+1/3=4/3;

W\_D=1+0/5=1

当t=5时，B完成，E已到达

W\_C=1+3/3=2;

W\_D=1+2/5=1.4;

W\_E=1+1/4=1.25

当t=8时，C完成

W\_D=1+5/5=2;

W\_E=1+4/4=2

综上所述，运行顺序为A->B->C->D->E 或者 A->B->C->E->D

2、有6个进程A、B、C、D、E、F，以及三种资源a、b、c，这6个进程对这三种资源的最大需求和已经分配的见下表，假如系统中现有a、b、c的数目分别为1,1,1，请按照能够避免死锁的银行家算法，给出相应的分配顺序，并且写出每次分配后，Work集合的变化。

|  |  |  |
| --- | --- | --- |
| 进程 | 最大需求  a、b、c | 已分配  a、b、c |
| A | 3，2，5 | 1，0，0 |
| B | 4，2，4 | 1，0，0 |
| C | 2，2，2 | 0，0，1 |
| D | 1，1，2 | 0，0，2 |
| E | 2，2，4 | 1，1，0 |
| F | 1，1，3 | 0，0，1 |

分配前，Work=[1，1，1]。

根据 最大需求-已分配<=当前Work 的原则：

首先符合条件的是进程D，

运行完毕后，Work=Work+Allocation[D]=[1，1，3]；

然后进程F符合条件，

运行完毕后，Work=Work+Allocation[F]=[1，1，4]；

然后进程E符合条件，

运行完毕后，Work=Work+Allocation[E]=[2，2，4]；

然后进程C符合条件，

运行完毕后，Work=Work+Allocation[C]=[2，2，5]；

然后进程A符合条件，

运行完毕后，Work=Work+Allocation[A]=[3，2，5]；

然后进程B符合条件，

运行完毕后，Work=Work+Allocation[B]=[4，2，5]；

因此，运行顺序为D、F、E、C、A、B。