Lista 2 – SO – Henrique Oliveira

Q1)

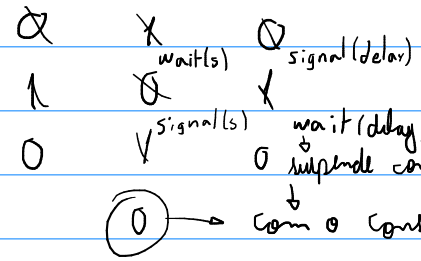
```
n : integer;
s, delay        : semaforo;  // binário

produtor {                                          consumidor {
        loop                                        wait(delay);
        {                                           loop {
                wait(s);                                wait(s);
                critico 1;                              critico 2;
                n = n + 1;                              n = n - 1;
                if (n = 1) then signal (delay);         if (n = 0) then wait(delay);
                signal(s);                              signal(s);
        }                                           }
}                                                   }

                        /// execução principal ///
        begin   n = 0;  s = 1;  delay = 0; cobegin   produtor; consumidor;  coend   end.
```

n	s	delay
0	1	0

→ começando c/ produtor
(mesmo que começasse c/ consumidor, wait(delay); suspenderia o consumidor)



(0) → com o consumidor suspenso, e com S=0, o programa entra em um impasse.

Para resolver esse problema, é preciso mover o trecho de comparação do if (if (n=0) then wait(delay);) para antes do wait(s);, para que então a checagem seja feita sem gerar um impasse no código.

Q2) Deadlock (impasse)

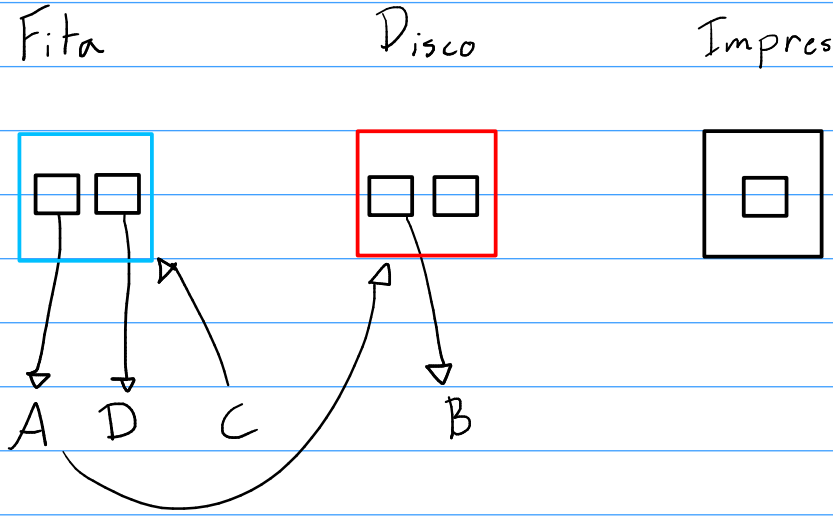
- Exclusão Mútua – apenas um processo por vez pode usar o recurso
- Posse e espera – processo aguarda o recurso usado por outro processo
- Não-preempção – recurso só pode ser liberado por quem o mantém
- Espera Circular – ciclo entre recursos e processos (trânsito)

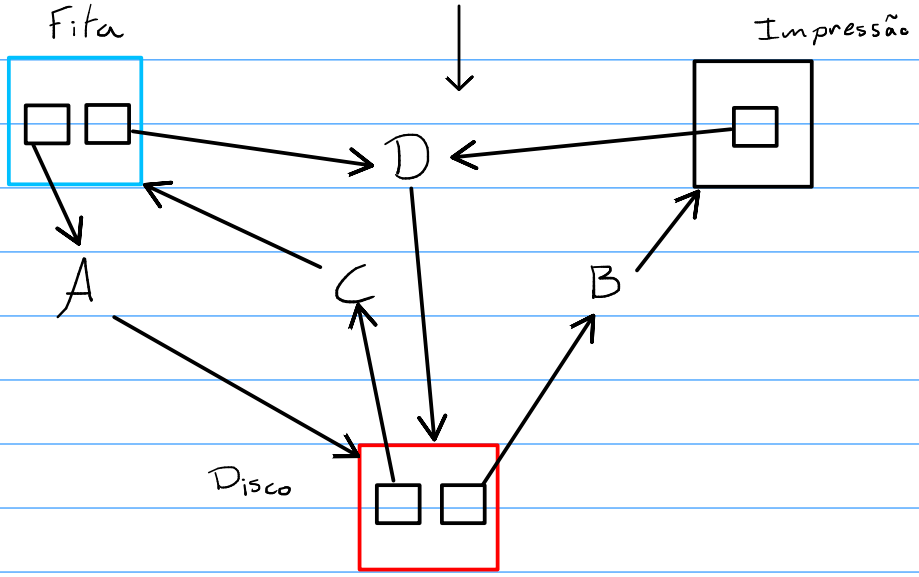
Q3) - 6 fitas
- N processos
- 2 fitas / processo

Para não haver deadlock, é necessário haver pelo menos uma fita liberada. Se 1 processo usa duas fitas, então o sistema é livre de Deadlocks quando N=1 (2 fitas ocupadas, 4 liberadas) e quando N=2 (4 fitas ocupadas, 2 liberadas). De 3 em diante, como não haverá mais fitas desocupadas, o sistema corre risco de Deadlock.

Q4)

Cenário Inicial:





A situação leva, sim, a um Deadlock. Isso ocorre pois todos os processos realizam requisições e estão de posse sobre uma ou mais instâncias dos recursos. Assim, observa-se as 4 condições que caracterizam Deadlocks, especialmente a de Espera Circular, no grafo acima. Logo, como todos os processos fazem uma requisição dependente de um recurso ocupado, e já que nenhum desses recursos serão liberados, há um impasse.