# Quarta Lista de Exercícios de Sistemas Operacionais

1. O código a seguir ilustra dois processos, A e B, que competem por dois recursos do sistema, 1 e 2. Existe a possibilidade de ocorrer deadlock no sistema? Justifique sua resposta.

```
semaphore resource_1;
semaphore resource_2;

void Process_A (void){
   down(&resource_1);
   down(&resource_2);
   use_both_resources();
   up(&resource_2);
   up(&resource_1);
}
 void Process_B (void){
   down(&resource_2);
   down(&resource_1);
   use_both_resources();
   up(&resource_1);
   up(&resource_2);}
```

```
semaphore resource_1;
semaphore resource_2;

void Process_A (void){
   down(&resource_1);
   down(&resource_2);
   use_both_resources();
   up(&resource_2);
   up(&resource_1);
}
 void Process_B (void){
   down(&resource_1);
   down(&resource_2);
   use_both_resources();
   up(&resource_1);
   up(&resource_2);}
```

2. Os códigos a seguir representam uma potencial solução para o problema do produtor/consumidor com buffer limitado. Analise o código apresentado e reposta as questões a seguir:

```
#define N 5                         /* number of philosophers */

void philosopher(int i)             /* i: philosopher number, from 0 to 4 */
{
     while (TRUE) {
          think( );                 /* philosopher is thinking */
          take_fork(i);             /* take left fork */
          take_fork((i+1) % N);     /* take right fork; % is modulo operator */
          eat( );                   /* yum-yum, spaghetti */
          put_fork(i);              /* put left fork back on the table */
          put_fork((i+1) % N);      /* put right fork back on the table */
     }
}
```

a. A solução apresentada utiliza as funções sleep() e wakeup(). Explique qual o propósito destas funções e como elas funcionam no código apresentado.
b. A utilização das primitivas sleep() e wakeup(), como apresentadas, elimina a possibilidade de ocorrência de deadlock no algoritmo? Caso positivo, justifique; caso negativo mostre como tal problema poderia ser contornado.

3. Desenhe o gráfico de acesso e calcule o número de deslocamentos entre cilindros de um disco para os algoritmos de escalonamento First-Come-First-Served, Shortest-Seek-Time-First e Scan (Elevator, com sentido inicial decrescente) para a seguinte fila de pedidos: 7, 144, 100, 20, 58, 15, 150, 75. O cabeçote de leitura encontra-se inicialmente na posição do cilindro 82 (última posição válida).

4. O controle de acesso a dispositivos de entrada e saída realizado pelo sistema operacional pode ser implementado através das técnicas de entrada e saída programada, entrada e
saída orientada a interrupção e entrada e saída com acesso direto à memória. Explique o funcionamento de cada um destes métodos.

5. Explique o que são os drivers de dispositivos e qual o seu papel na comunicação com os controladores de dispositivos de entrada e saída.

6. Qual a função básica de um driver de dispositivo? Exemplifique sua interação com os outros elementos de um sistema computacional.

7. Explique o que são e a serventia de interrupções de entrada e saída. Dê um exemplo de como interrupções ocorrem durante a execução de um programa na UCP.

8. O controle de acesso a dispositivos de entrada e saída realizado pelo sistema operacional pode ser implementado através das técnicas de entrada e saída programada, entrada e saída orientada a interrupção e entrada e saída com acesso direto à memória. Caracterize o funcionamento de cada um destes métodos e explique sua influência no desempenho do sistema computacional.

9. Descreva o que representam as seguintes situações em um sistema computacional e diga qual a relação delas com a ocorrência de uma situação de deadlock.
a. Exclusão mútua;
b. Posse e espera;
c. Sem preempção;
d. Espera circular;