Sistemas Operacionais
Segunda Prova - 01/06/2023

NOME: _____

1. Explique como um certo conjunto de processos concorrentes entra em *deadlock* em algumas execuções (um terço delas) e opera normalmente em outras. O que pode ser tratado de forma a evitar a ocorrência de *deadlocks*? (VALOR 2 pontos)

   É uma situação que depende da ordem de escalonamento dos processos, que pode estar evitando parte das ocorrências de *deadlocks*.

   Isso pode ser corrigido alterando-se os algoritmos usados nos processos, buscando-se eliminar o conflito que leva aos eventuais *deadlocks*.

2. Um projetista de um S.O. pensa em usar o algoritmo do elevador para gerenciamento de um SSD. O que pode justificar essa escolha? (VALOR 1,5 pontos)

   Considerando que em SSDs as páginas estão arranjadas e são acessadas por endereços, aplicar o algoritmo do elevador faz algum sentido apenas para operações de escrita, quando poderia haver algum agrupamento mais eficiente de blocos e páginas.

   Para operações de leitura não faz sentido algum usar outro critério que não seja FIFO.

3. Considerando E/S para disco, qual seria a vantagem em acumular, quando possível, várias operações para trilhas próximas para serem feitas de uma vez? (VALOR 2 pontos)

   A vantagem aparece na otimização dos tempos de latência, com a possível serialização no acesso a setores conscutivos das trilhas próximas entre si.

4. Aplicações que demandam muito acesso a disco necessitam, de mecanismos alternativos para gerenciamento da CPU. Que parâmetros devem ser considerados ao especificar o escalonador de processos num sistema que tenha predominância de aplicações desse tipo? (VALOR 1,5 pontos)

   O escalonador deve priorizar, de algum modo, processos que tenham acabado de fazer E/S, pois teoricamente teriam mais tempo de processamento antes da próxima operação de E/S.

5. Uma aplicação concorrente tem os processos A, B e C, sendo que os processos A e B consomem valores produzidos pelo processo C especificamente para cada um deles, a partir de um valor produzido pelo processo A (antes de consumir o valor produzido por C. O processo C produz sempre um valor para o processo A, porém produz valores para o processo B apenas quando A produz um valor maior que X. Além disso, o processo B só volta a consumir quando um novo valor for produzido para ele. Escreva então protótipos desses processos usando semáforos. (VALOR 3 pontos)

```
CRONOLOGIA DE AÇÕES
        Processo A produz algo (valA) para consumo de processo C e sinaliza isso.
        Processo C consome o que foi produzido e produz resposta ao process A e sinaliza
        Processo C produz algo para processo B se valA > X (e sinaliza para B)
        Processo A consome o que processo C produziu para ele)
        Processo B consome, quando for o caso, o que processo C produziu para ele

process A {
    while (true) {
        faz_algo();
        produz(valA);
        V(semC); //sinaliza que produziu
        P(semA); //espera que C produza
        consome(respCA);
    }
}

process B {
    while (true) {
        faz_algo();
        P(semB); //espera sinalização do processo B
        consome(respCB);
    }
}

process C {
    while (true) {
        faz_algo();
        P(semC);
        consome(valA);
        produz(respCA);
        V(semA); //sinaliza ao processo A
        if (valA > X)
            { produz(respCB);
              V(semB); //sinaliza ao processo B quando for o caso
            }
    }
}
```

Boa prova,