Curso de Tecnologia em Sistemas de Computação
Disciplina de Sistemas Operacionais
Professores: Valmir C. Barbosa e Felipe M. G. França
Assistente: Alexandre H. L. Porto

Quarto Período
Gabarito da AP1 - Primeiro Semestre de 2017

Nome -
Assinatura -

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

---

1. (1,5) Suponha que um programa A, que requer 90ms de execução no processador, faça somente uma operação de E/S. Suponha ainda que tal operação faça o processador ficar ocioso 25% do tempo total de execução de A. Suponha agora que um programa B, que não faz operações de E/S, deva compartilhar o processador com A. Qual deverá ser o tempo mínimo de execução de B no processador para que o processador não fique ocioso? Suponha que a multiprogramação somente é usada para evitar a ociosidade do processador quando operações de E/S são feitas. Justifique a sua resposta.

   Resp.: Pelo enunciado, o tempo total de execução do programa A é de 90ms (o tempo de execução do programa no processador) mais o tempo t para fazer a única operação de E/S (tempo este que é o responsável pela ociosidade do processador). Como o tempo da operação de E/S é igual a 25% do tempo total de execução, então $0,25 \times (90 + t) = t$, ou seja, $t = 30$ms. Logo, para que um programa B, que não faz operações de E/S, evite a ociosidade do processador, basta que o seu tempo de execução total, que é igual ao tempo de execução no processador, seja maior ou igual a 30ms.

2. (2,5) Diga se as seguintes afirmativas são falsas ou verdadeiras. Para responder, escreva apenas F ou V para cada item em seu caderno de respostas.

   (a) (0,5) Antes do desenvolvimento dos sistemas em lote, os operadores eram os responsáveis pelo gerenciamento dos programas submetidos pelos usuários, que depois precisavam esperar os operadores retornarem os resultados dos programas.

   Resp.: V (Verdadeira).

   (b) (0,5) Um processo é um arquivo especial do disco usado para representar um programa em execução no sistema operacional.

   Resp.: F (Falsa), pois um processo é somente uma representação de um programa em execução no sistema operacional, não sendo

portanto um arquivo especial do sistema de arquivos.

(c) (0,5) A multiprogramação sempre é visível ao usuário do sistema operacional, independentemente de máquinas virtuais ou exonúcleos serem usados ao gerenciá-la.

Resp.: F (Falsa), pois a multiprogramação não é visível ao usuário quando uma máquina virtual ou um exonúcleo são usados para gerenciá-la.

(d) (0,5) Um processo passa do estado **Executando** para o estado **Bloqueado** quando ele termina a sua execução, ficando neste estado até que seja novamente executado, quando ele passará ao estado **Pronto** e ali permanecerá até ser escolhido pelo escaonador.

Resp.: F (Falsa), pois as transições dadas no enunciado não correspondem àquelas vistas na aula 4.

(e) (0,5) O conceito de exclusão mútua evita as condições de corrida, porque impede que dois ou mais processos usem simultaneamente um recurso compartilhado por eles.

Resp.: V (Verdadeira).

3. (1,5) Diga a quais conceitos vistos em aula se referem as seguintes definições:

(a) (0,5) Nome dado ao arquivo usado para acessar um dispositivo que pode ser modelado por um fluxo de caracteres.

Resp.: Dispositivo de caracteres.

(b) (0,5) Nome do contador que sempre bloqueia um processo que tente decrementá-lo quando é igual a 0, sendo que neste caso o

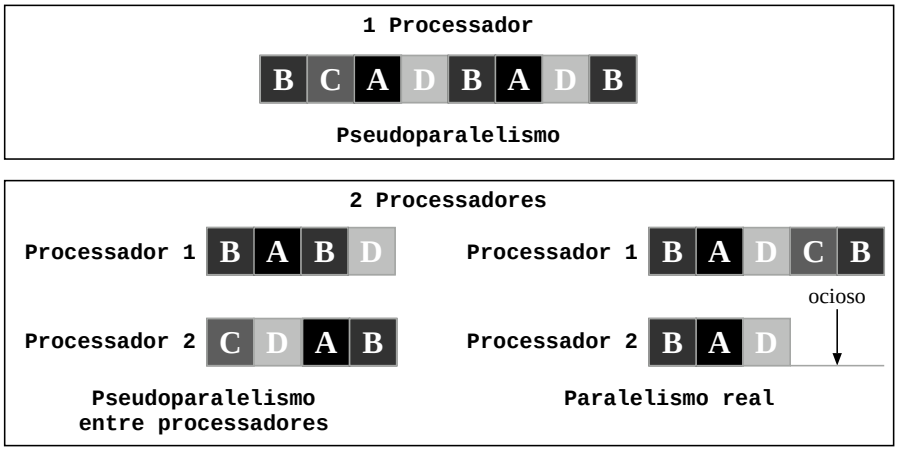
processo é colocado em uma fila de processos bloqueados até que algum outro processo tente incrementar o contador, o que somente será possível quando todos os processos da fila tiverem sido desbloqueados.

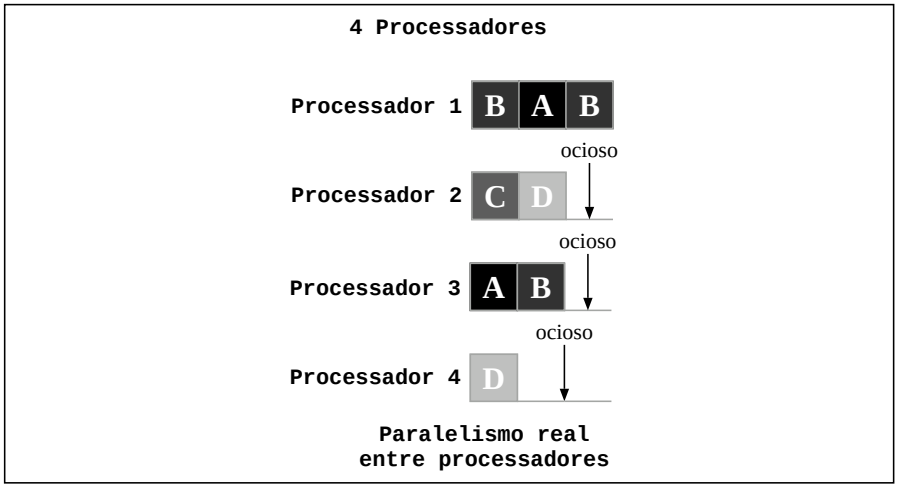
Resp.: Semáforo.

(c) (0,5) Nome dado ao invervalo de tempo durante o qual um processo pode executar, quando o algoritmo *round robin* é usado, antes de o processo ser suspenso e colocado no estado **Pronto**.

Resp.: Quantum.

4. (1,5) Um aluno de sistemas operacionais afirmou que cada ordem de execução e tipo de paralelismo dados na figura a seguir estão corretos. Todas as afirmações do aluno estão corretas? Se você acha que sim, basta dizer isso mas, se você acha que não, diga quais afirmações estão erradas.





Resp.: Duas das afirmações do aluno não estão corretas, porque existem dois erros na figura que mostra a execução com 2 processadores. Na parte esquerda da figura, temos um paralelismo real entre processadores. O pseudoparalelismo somente ocorre quando existe um único processador no sistema. Já na parte direta da figura, o problema ocorre devido a um mesmo processo estar executando concorrentemente em mais de um processador. Se fosse possível dividir um processo em partes que pudessem ser executadas em paralelo, essas partes seriam executadas por processos diferentes.

5. (1,5) Suponha que dois processos, A e B, compartilhem uma árvore, inicialmente sem nós, que pode armazenar até n nós. O processo A continuamente cria um novo nó e o associa aleatoriamente a um dos nós da árvore, se a árvore não estiver cheia, enquanto que o processo B sempre remove um dos nós folhas da árvore, se a árvore não estiver vazia. Como um semáforo binário e dois semáforos de contagem podem ser usados para garantir o correto funcionamento dos processos A e B? Justifique a sua resposta.

Resp.: A seguir mostramos como três semáforos, um binário e dois de contagem, podem ser usados para implementar os códigos dos processos. O semáforo binário, chamado *acesso*, é usado para garantir o acesso exclusivo à arvore. O primeiro semáforo de contagem, chamado

vazias, conta o número de nós que ainda podem ser criados na árvore e é usado para bloquear o processo A quando a árvore está cheia. Finalmente, o segundo semáforo de contagem, chamado *cheias*, conta o número nós da árvore e é usado para bloquear o processo B quando a árvore está vazia. Como inicialmente a árvore está vazia e não está sendo usada, então os semáforos *vazias*, *cheias* e *acesso* são inicializados, respectivamente, com n, 0 e 1. A seguir mostramos os códigos para os processos A e B, sendo que a função $removenofolha()$ remove e retorna um dos nós folha da árvore, e a função $insereno(e)$ insere o nó e na árvore:

```
void ProcessoA(void)
{
  while(1)
  {
    // Código para criar um novo nó e depois salvá-lo em e.
    // Usa a operação P sobre vazias para garantir que podemos criar um novo
    // nó na árvore.
    P(vazias);
    // Garante o acesso exclusivo à árvore.
    P(acesso);
    // Insere e na árvore.
    insereno(e);
    // Libera o acesso exclusivo à árvore.
    V(acesso);
    // Usa a operação V sobre cheias para registrar que um nó foi inserido na
    // árvore.
    V(cheias);
  }
}
```

```
void ProcessoB(void)
{
  while(1)
  {
    // Usa a operação P sobre cheias para garantir que existe pelo menos um nó
    // folha na árvore.
    P(cheias);
    // Garante o acesso exclusivo à árvore.
    P(acesso);
    // Remove um nó folha da árvore e o armazena em e.
    e = removenofolha();
    // Libera o acesso exclusivo à árvore.
    V(acesso);
    // Usa a operação V sobre vazias para registrar que um nó folha foi removido
    // da árvore.
    V(vazias);
    // Código para usar o nó folha e.
  }
}
```

6. (1,5) Suponha que três processos, A, B e C, precisem executar no processardor por, respectivamente, 8ms, 5ms e 4ms. Se o algoritmo de *round robin* for usado ao escalonar os processos, com um quantum de 4ms, que ordem inicial de execução dos processos A, B e C vai gerar a menor soma dos tempos de término dos processos? Justifique a sua resposta.

   Resp.: Existem seis possíveis ordens de execução, dadas nas tabelas a seguir. Nestas tabelas mostramos como os processos são escolhidos pelo algoritmo, sendo que cada coluna refere-se à execução de um processo, dando o tempo de início de cada quantum e o processo correspondente. A ordem inicial usada pela execução dada em cada tabela é definida, da esquerda para a direita, pelos processos nas três colunas iniciais da tabela. Além disso, na última linha de cada tabela, damos a soma dos tempos de término, em ordem, de A, B e C, para a ordem inicial definida pela tabela. Devido ao tempo do processo B não ser múltiplo do tamanho do quantum de 4ms, B somente usa 1ms do seu último quantum. Pelas tabelas, a ordem com a menor soma para os tempos, de 34ms, é C, B, A.

0	4	8	12	16
A	B	C	A	B
16+17+12=45ms				

0	4	8	12	16
A	C	B	A	B
16+17+8=41ms				

0	4	8	12	13
B	A	C	B	A
17+13+12=42ms				

0	4	8	12	13
B	C	A	B	A
17+13+8=38ms				

0	4	8	12	16
C	A	B	A	B
16+17+4=37ms				

0	4	8	12	13
C	B	A	B	A
17+13+4=34ms				

Na verdade, podemos notar que a menor soma dos tempos de término é obtida quando a ordem inicial de execução dos processos é igual à ordem crescente de seus tempos.