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 AD1 - Primeiro Semestre de 2017

Atenção: Cada aluno é responsável por redigir suas próprias respostas. Provas iguais umas às outras terão suas notas diminuídas. As diminuições nas notas ocorrerão em proporção à similaridade entre as respostas. Exemplo: Três alunos que respondam identicamente a uma mesma questão terão, cada um, 1/3 dos pontos daquela questão.

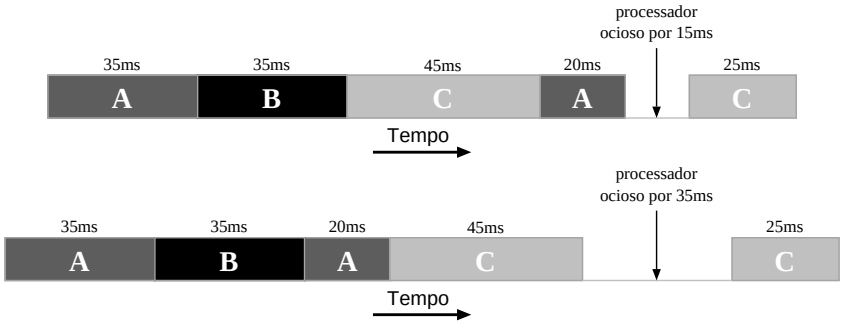
Nome -
Assinatura -

---

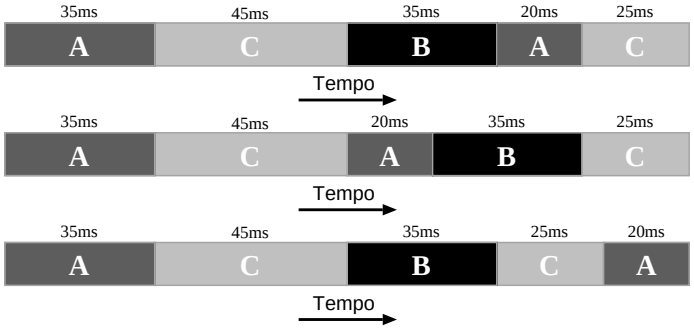
1. (1,0) Suponha que três programas, A, B e C, precisem executar no processador por, respectivamente, 55ms, 35ms e 70ms. Suponha ainda que A e C façam somente uma operação de E/S, e que B não faça operações de E/S. Se A faz a sua operação de E/S, com duração de 25ms, após executar por 35ms no processador, se C faz a sua operação de E/S, com duração de 35ms, após executar por 45ms no processador, e se a multiprogramação é usada somente para evitar a ociosidade do processador quando operações de E/S são feitas, que ordem inicial de execução dos programas A, B e C gera a menor ociosidade do processador? Justifique a sua resposta.

Resp.: Existem seis possíveis ordens iniciais de execução para os programas A, B e C. Elas dão dadas nas figuras a seguir. Como podemos ver, para que o processador não fique ocioso, não podemos usar as ordens iniciais A, B, C ou B, A, C.

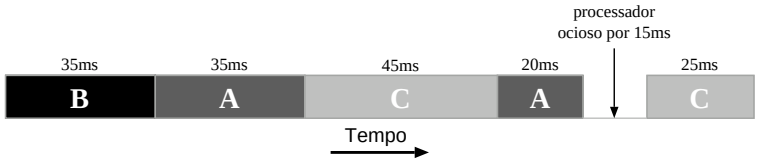
A, B, C:



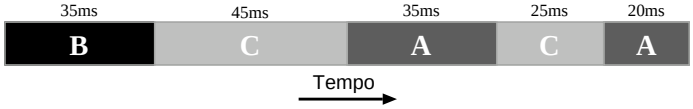
A, C, B:



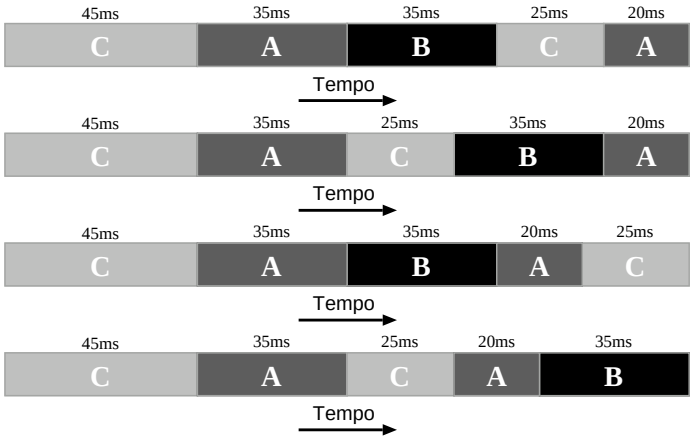
B, A, C:



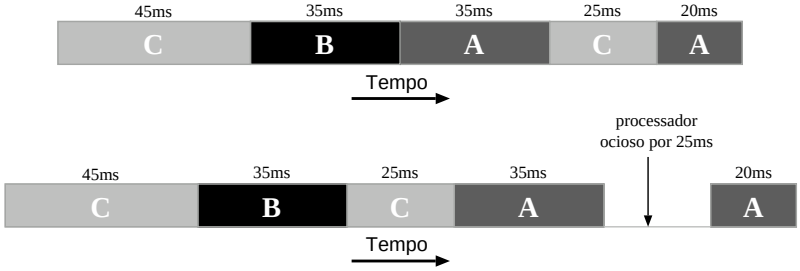
B, C, A:



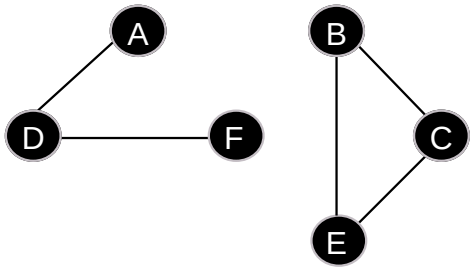
C, A, B:



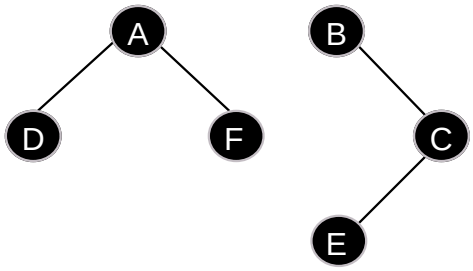
C, B, A:



2. (1,0) Um aluno de sistemas operacionais alegou que a hierarquia dada a seguir relaciona os processos A, B, C, D, E e F, em execução no sistema operacional. A alegação do aluno está correta? Justifique a sua resposta.



Resp.: A alegação do aluno não está correta porque existem dois erros na figura. O primeiro erro é que, estando no mesmo nível, D e F não podem estar relacionados um com o outro. O segundo erro é que a aresta entre B e E não faz sentido, já que o pai de E tem que estar em um nível imediatamente acima dele. Uma possível alternativa à figura é a seguinte:



3. (1,5) Suponha que o sistema operacional esteja executando diretamente sobre o hardware de um computador cujas operações de E/S demorem t ms. Suponha ainda que um processo tenha executado por 9,5s e que, durante a sua execução, tenha feito 4 000 operações de E/S. Se o sistema operacional agora executar sobre uma máquina virtual que reduza a velocidade do processador em 45% e a velocidade das operações de E/S em 60%, que valor de t fará o processo executar na máquina virtual por 15s fazendo 2 000 operações de E/S? Justifique a sua resposta.

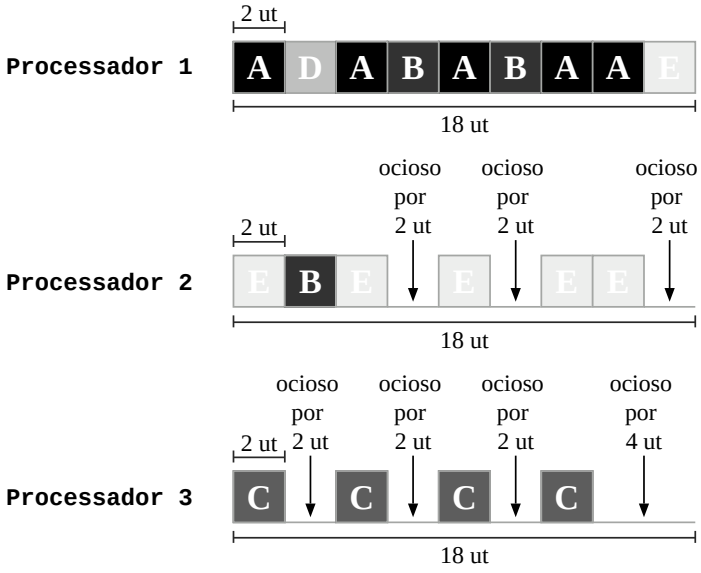
   Resp.: Como o tempo total de execução é de 9,5 s ou 9 500 ms, e como o processo faz 4 000 operações de E/S, então $4\,000t$ ms dos 9 500 ms são gastos com operações de E/S, quando a execução ocorre sobre

o hardware do computador. Logo, o processo executa no processador do hardware por $9\,500 - 4\,000t$ ms. Note que a velocidade do processador ser reduzida em 45% significa que a velocidade do processador virtual é 55% da velocidade do processador real, o que por sua vez significa que, durante os $9\,500 - 4\,000t$ ms, somente 55% das instruções são executadas. Com isso, quando o processo executa sobre a máquina virtual, seu tempo de execução no processador virtual é de $\frac{9\,500-4\,000t}{0,55}$ ms. Agora, como o processo executa 2 000 operações de E/S na máquina virtual, e como o novo tempo de cada operação de E/S é de $\frac{t}{0,4} = 2,5t$ ms (já que, similarmente à redução do tempo do processador, a redução da velocidade de cada operação de E/S em 60% significa que no mesmo tempo podemos, na máquina virtual, executar somente 40% das operações de E/S originais), então $2\,000 \times 2,5t = 5\,000t$ ms dos 15s ou 15 000 ms do tempo de execução do processo na máquina virtual são gastos com E/S. Logo, o tempo de execução do processo no processador virtual é de $15\,000 - 5\,000t$ ms e, com isso, podemos concluir que $\frac{9\,500-4\,000t}{0,55} = 15\,000 - 5\,000t$, ou seja, que o tempo t da operação de E/S no hardware real é de 1 ms.

4. (1,5) Suponha que o escalonador sempre substitua o processo em execução em um processador a cada 2 unidades de tempo, e que um processo somente possa executar novamente quando cada um dos outros processos prontos tiver sido executado também por 2 unidades de tempo em algum processador. Suponha ainda que cinco processos, A, B, C, D e E, que não são bloqueados durante a sua execução, precisem executar no processador por, respectivamente, 10, 6, 8, 2 e 12 unidades de tempo. Se a ordem inicial de execução dos processos for A, E, C, D e B, qual será o tempo decorrido após a execução de todos os processos se somente 1 processador existir no hardware? E se existirem 3 processadores? E se existirem 5 processadores? Justifique a sua resposta.

Resp.: -Se existir somente 1 processador, então o tempo decorrido será igual ao tempo necessário para executar todos os cinco processos e não dependerá do algoritmo de escalonamento usado ou da ordem inicial de execução dos processos. Logo, o tempo total de execução será $10 + 6 + 8 + 2 + 12$ unidades de tempo, ou seja, 38 unidades de tempo.

-Agora, se existirem 3 processadores, 3 processos poderão executar paralelamente. Mais de uma solução resultando no menor tempo total possível poderá existir, por exemplo, a ordem de execução dada na figura a seguir, na qual para simplificar usamos "ut" para representar 1 unidade de tempo. Como podemos ver pela figura, o tempo total de execução de A, B, C, D e E será de 18 unidades de tempo.



-Finalmente, no caso de 5 processadores, podemos supor, sem perda de generalidade, que cada processo executará em um dos processadores até terminar. Com isso, o tempo total de execução de A, B, C, D e E será o maior tempo de execução de um desses processos, ou seja, 12 unidades de tempo.

5. (2,0) Suponha que três processos, A, B e C, compartilhem uma fila e uma pilha, ambas inicialmente vazias, e que a fila possa armazenar n elementos e a pilha um número ilimitado de elementos. O processo A continuamente adiciona um novo elemento no final da fila se ela não está cheia. Já o processo B continuamente move, se a fila possui pelo menos dois elementos, os dois elementos no início da fila para a pilha,

sendo que o elemento no início da fila estará, após os dois elementos terem sido movidos, no topo da pilha. Finalmente, o processo C remove o elemento do topo da pilha se a pilha não está vazia. Como dois semáforos binários e três semáforos de contagem podem ser usados para garantir o correto funcionamento dos processos A, B e C? Justifique a sua resposta.

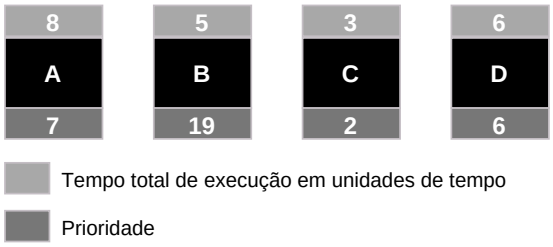
Resp.: A seguir mostramos como os cinco semáforos, dois binários e três de contagem, podem ser usados para implementar os códigos dos processos. O primeiro semáforo binário, chamado $acesso_F$, é usado para garantir o acesso exclusivo à fila. O segundo semáforo binário, chamado $acesso_P$, é usado para garantir o acesso exclusivo à pilha. O primeiro semáforo de contagem, chamado $vazia_F$, conta o número de entradas não usadas por elementos na fila e é usado para bloquear o processo A quando a fila está cheia. O segundo semáforo de contagem, chamado $cheia_F$, conta o número de entradas com elementos na fila e é usado para bloquear o processo B quando a fila está vazia. Finalmente, o terceiro semáforo de contagem, chamado $cheia_P$, conta o número de elementos na pilha e é usado para bloquear o processo C quando a pilha está vazia. Como inicialmente a fila e a pilha estão vazias e não estão sendo usadas, então os semáforos $vazia_F$, $cheia_F$, $cheia_P$, $acesso_F$ e $acesso_P$ serão inicializados, respectivamente, com n, 0, 0, 1 e 1. A seguir mostramos os códigos para os processos A, B e C, sendo que a função $insere fila(e)$ insere o elemento e no final da fila, que a função $remove fila()$ remove e retorna o elemento do início da fila, que a função $insere topo(e)$ insere o elemento e no topo da pilha, e que a função $remove topo()$ remove e retorna o elemento no topo da pilha:

```
void ProcessoA(void)
{
  while(1)
  {
    // Código para gerar um elemento e salvá-lo em e.
    // Usa a operação P sobre vazia_F para garantir que pelo menos uma
    // entrada esteja livre na fila.
    P(vazia_F);
    // Garante o acesso exclusivo à fila.
    P(acesso_F);
    // Insere e no final da fila.
    inserefila(e);
    // Libera o acesso exclusivo à fila.
    V(acesso_F);
    // Usa a operação V sobre cheia_F para registrar que um elemento foi
    // inserido na fila.
    V(cheia_F);
  }
}
```

```
void ProcessoB(void)
{
  while(1)
  {
    // Usa a operação P sobre cheia_F para garantir que pelo menos dois
    // elementos existam na fila.
    P(cheia_F);
    P(cheia_F);
    // Garante o acesso exclusivo à fila.
    P(acesso_F);
    // Remove dois elementos do início da fila e os armazena em e_1 e e_2.
    e_1 = removefila();
    e_2 = removefila();
    // Libera o acesso exclusivo à fila.
    V(acesso_F);
    // Usa a operação V sobre vazia_F para registrar que dois elementos
    // foram removidos da fila.
    V(vazia_F);
    V(vazia_F);
    // Garante o acesso exclusivo à pilha.
    P(acesso_P);
    // Insere os dois elementos e_1 e e_2 na pilha, garantindo que e_1 fique
    // no topo e e_2 no subtopo.
    inseretopo(e_2);
    inseretopo(e_1);
    // Libera o acesso exclusivo à pilha.
    V(acesso_P);
    // Usa a operação V sobre cheia_P para registrar que dois elementos
    // foram inserido na pilha.
    V(cheia_P);
    V(cheia_P);
  }
}
```

```
void ProcessoC(void)
{
  while(1)
  {
    // Garante que existe pelo menos um elemento na pilha.
    P(cheia_P);
    // Garante o acesso exclusivo à pilha.
    P(acesso_P);
    // Remove o elemento do topo da pilha e o armazena em e.
    e = removetopo();
    // Libera o acesso exclusivo à pilha.
    V(acesso_P);
    // Código para usar o elemento e.
  }
}
```

6. (3,0) Suponha que os processos da figura dada a seguir tenham acabado de entrar no estado pronto, e que sejam os únicos processos em execução. Quais seriam os tempos de término desses processos se o sistema operacional usasse, ao escalonar os processos:



   (a) (1,0) o algoritmo de *round robin*, supondo um quantum de 2 unidades de tempo e que a ordem inicial de execução dos processos seja A, D, C e B.

   Resp.: Pelo enunciado, vemos que a ordem de execução dos processos é como dado na tabela a seguir. Nesta tabela 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. Devido a os tempos dos processos B e C não serem múltiplos do tamanho do quantum de 2 unidades de tempo, B e C somente usam 1 unidade

de tempo do seu último quantum. Pela tabela, vemos que os tempos de término dos processos A, B, C e D, são de, respectivamente, 22, 20, 13 e 19 unidades de tempo.

0	2	4	6	8	10	12	13	15	17	19	20
A	D	C	B	A	D	C	B	A	D	B	A

(b) (1,0) o algoritmo de prioridades, supondo que a prioridade é incrementada em 3 unidades a cada 2 unidades de tempo e que o processo em execução continue a executar até existir um outro processo com prioridade menor do que a dele.

Resp.: A tabela a seguir é similar à tabela do item anterior e possui somente mais uma linha, a terceira, que mostra a prioridade de cada processo antes de ele executar na unidade de tempo dada na mesma coluna. Pela tabela, vemos que os tempos de término dos processos A, B, C e D, são de, respectivamente, 17, 22, 3 e 13 unidades de tempo.

0	2	3	5	7	9	11	13	15	17	19	21
C	C	D	A	D	A	D	A	A	B	B	B
2	5	6	7	9	10	12	13	16	19	22	25

(c) (1,0) o algoritmo do trabalho mais curto primeiro.

Resp.: Como vimos na aula 6, no algoritmo do trabalho mais curto primeiro, os processos são executados, em ordem crescente, de acordo com os seus tempos de execução. Além disso, quando um processo começa a executar, ele executa exclusivamente no processador até terminar. Então a única possível ordem de execução é C, B, D e A. O processo C então executa do tempo 0 até o 3, o processo B do tempo 3 até o 8, o processo D do tempo 8 até o 14 e, finalmente, o processo A do tempo 14 até o 22. Logo, os tempos de término dos processos A, B, C e D são de, respectivamente, 22, 8, 3, e 14 unidades de tempo.