I Lista de Exercícios de Sistemas Operacionais

Prof. André Leon S. Gradvohl, Dr.

gradvohl@ft.unicamp.br

1ª Questão

Em relação ao *kernel*, explique:

a) O que é o *kernel* do Sistema Operacional e defina os tipos de *kernel* (monolítico, em camadas e microkernel).

b) Por que a tabela de processos é necessária em um sistema de tempo compartilhado? Ela também é necessária em sistemas de computadores pessoais, nos quais existe apenas um processo, que detém o comando de toda a máquina até que ele termine? Justifique.

2ª Questão

a) Explique, em poucas palavras, o que é multiprogramação.

b) É possível implementar multiprogramação, mesmo com uma CPU com um único núcleo?

3ª Questão

1) Em laços do tipo for, qual a vantagem de usar registradores para armazenar as variáveis contadoras do laço ao invés de usar a memória RAM? Explique.

2) Considere o código a seguir escrito em Assembly:

```
1:  mov x, 10h
2:  mov y, 20h
3:  cmp x, y
4:  jbe 7
5:  add x, 1h
6:  jmp 8
7:  sub y, 1h
8:  mov z, 0h
9:  add z, x
10: add z, y
```

Começando da primeira instrução, para cada ciclo de instrução (→) escreva o conteúdo dos dois registradores especiais - Contador de Programas (CP) e Registrador de Instruções (RI).

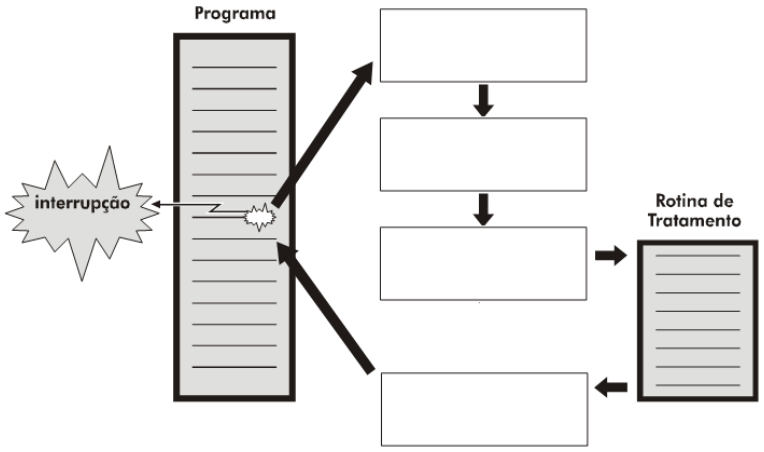
4ª Questão

No UNIX, usando-se o comando `fork( )`, um novo processo é criado sem que um novo programa seja posto em execução. Assinale **V** (verdadeiro) ou **F** (falso) para as afirmações a seguir:

( )	Caso não ocorra falha, a chamada no sistema fork( ), retorna no mesmo instante o valor do PID do processo filho para o processo pai.
( )	Se um processo B for criado por um processo A via fork( ), o processo B executará o programa que estiver sendo executado por A repetindo exatamente os mesmos fluxos.
( )	A mudança de variável global em um processo filho afeta o processo pai, pois os dados dos dois processos são iguais.
( )	Os valores iniciais das variáveis do processo filho são iguais às do processo pai no momento da execução fork( ).

5ª Questão

Explique o que significa a figura a seguir e preencha os quadros em branco com as frases à direita.



restaura o conteúdo dos registradores

obtém o endereço da rotina de tratamento

identifica a origem do evento

salva conteúdo dos registradores na pilha de controle

6ª Questão

Suponha 3 processos (P_1, P_2 e P_3), com um único *thread* cada e com as seguintes características. 50% do tempo do processo P_1 é gasto com operações de entrada e saída (E/S); 70% do processo P_2 é gasto com operações de E/S; e 80% do tempo do processo P_3 é gasto com E/S. Em função disso, ao longo de 10 unidades de tempo, quanto tempo uma CPU com um único núcleo fica ociosa? Se a CPU tivesse dois núcleos, quanto tempo essa CPU ficaria totalmente ociosa ao longo de 10 unidades de tempo, considerando o mesmo cenário?

7ª Questão

Thread é uma linha de execução dentro de um processo. Multithread são várias linhas de execução dentro do mesmo processo. Dos itens abaixo, quais são e quais não são compartilhados por todos os threads dentro do mesmo processo. Ligue a coluna da esquerda com a direita

- Variáveis globais
- Espaço de endereçamento
- Contador de programa
- Arquivos abertos
- Registradores
- Processos filhos
- Variáveis locais
- Pilha
- Estado
- Sinais e manipuladores de sinais

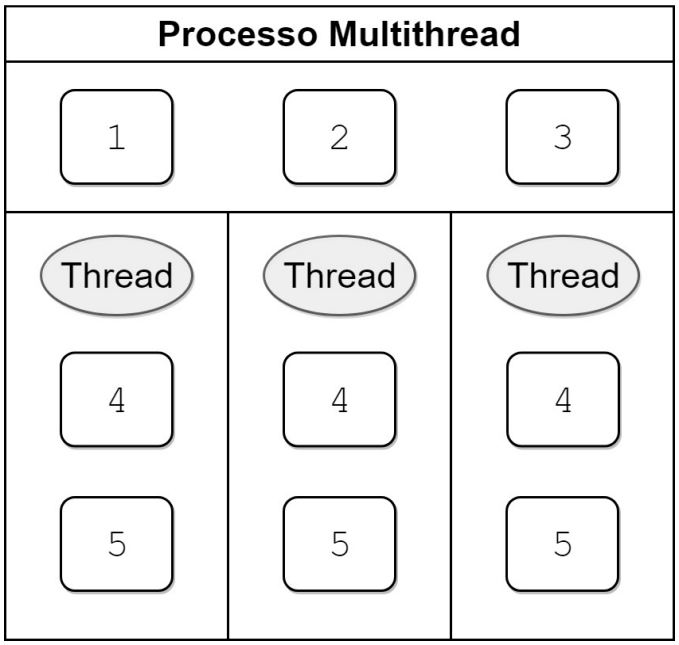
Itens <u>compartilhados</u> por todos os threads de um mesmo processo

Itens <u>não compartilhados</u> por todos os threads de um

8ª Questão

A figura a seguir mostra um processo com múltiplos threads. Os itens 1, 2 e 3 são associados ao processo e compartilhados entre as os threads. Entretanto, cada thread possui acesso exclusivo aos itens 4 e 5. Assinale a alternativa que associa corretamente os itens 1, 2, 3, 4 e 5 respectivamente:

a) Área de Código, Pilha, Registradores, Variáveis Globais e Ponteiros de Arquivos.
b) Registradores, Ponteiros de Arquivos, Área de Código, Pilha e Variáveis Globais.
c) Área de Código, Pilha, Variáveis Locais, Registradores e Ponteiros de Arquivos.
d) Variáveis Globais, Área de Código, Ponteiros de Arquivos, Pilha e Registradores.
e) Pilha, Variáveis Locais, Ponteiros de Arquivos, Registradores e Área de Código.



9ª Questão

Observe-se na tabela a seguir os processos, seus respectivos tempos de execução (ut) e suas prioridades:

Processo	Tempo de execução	Prioridade
P_0	4 ut	3
P_1	6 ut	2
P_2	3 ut	2
P_3	4 ut	1
P_4	2 ut	1

Suponha que no tempo t=4 chegue o processo P_5, com prioridade 2 e tempo de execução 3 ut.

Informe, para cada processo, qual o tempo total da sua execução para cada tipo de escalonamento adotado:

a) FIFO.
b) SJF.
c) Round Robin (quantum = 3 ut).
d) Fila de Prioridades.

10ª Questão

As regras de uso de um banheiro permitem que, quando uma mulher estiver no banheiro, outra mulher poderá entrar, mas um homem não e vice-versa. Implemente uma solução computacional para esse problema, supondo vários processos (homens e mulheres) e usando semáforos através das funções `void mulher_quer_entrar(int idM)` e `void homem_quer_entrar(int idH)`. Suponha que as seguintes funções já estão implementadas:

- `h_entra(int idH)`: função que mapeia a entrada de um homem no banheiro.
- `m_entra(int idM)`: função que mapeia a entrada de uma mulher no banheiro.
- `h_sai(int idH)`: função que mapeia a saída de um homem no banheiro.
- `m_sai(int idM)`: função que mapeia a saída de uma mulher do banheiro.

11ª Questão

Existem **n** passageiros que repetidamente aguardam para entrar em um carrinho da montanha russa, fazem o passeio e voltam a aguardar. Vários passageiros podem entrar no carrinho ao mesmo tempo, pois este tem várias portas. A montanha russa tem somente um carrinho, onde cabem **C** passageiros (**C** < **n**). O carrinho só começa seu percurso se estiver lotado.

Faça um programa que, usando semáforos, resolva o problema da liberação do carrinho e que faça os passageiros aguardarem, caso o carrinho esteja lotado.

Implemente as seguintes funções:

- `passageiro(int id)`: função que simula o comportamento do passageiro.
- `carrinho( )`: função que simula o comportamento do carrinho.

Declare todos os semáforos e contadores necessários para o seu programa. Use a função a seguir, se necessário:

- `libera_carrinho()`: função que libera o carrinho para dar uma volta.

12ª Questão

A situação a seguir representa o estado de um sistema computacional com vários recursos de cada tipo. Preencha o vetor E, substituindo as variáveis X, Y, V e W pelos valores correspondentes. Em seguida determine se o sistema está em *deadlock* ou não. Caso não esteja, verifique em que ordem os processos devem ser atendidos. Os processos são nomeados de P_0 a P_4 e os recursos são nomeados de R_0 a R_3.

$$E = (X \quad Y \quad V \quad W); \quad A = (1 \quad 1 \quad 1 \quad 4)$$

$$C = \begin{pmatrix} 1 & 1 & 1 & 2 \\ 4 & 0 & 1 & 2 \\ 1 & 1 & 3 & 0 \\ 3 & 5 & 2 & 0 \\ 2 & 1 & 2 & 0 \end{pmatrix}; \quad R = \begin{pmatrix} 9 & 5 & 0 & 0 \\ 4 & 0 & 3 & 0 \\ 7 & 5 & 2 & 3 \\ 0 & 1 & 0 & 2 \\ 10 & 2 & 3 & 5 \end{pmatrix}$$

13ª Questão

Em um sistema com multiprogramação, os recursos que podem ser usados por 5 programas que serão executados “simultaneamente” são relacionados na tabela a seguir:

Processo	Recursos previstos	Recursos alocados
P_0	a, e , j, g	j
P_1	a, b, f, g, h	a, f
P_2	c, d, h, i	c, h
P_3	b, c, g, k	b, g
P_4	d,e,i,k	d, k

Os recursos “e” e “i” estão disponíveis.

Pergunta-se:

a) O recurso “i” pode ser atribuído ao processo P_4 sem risco de *deadlock*? Justifique.

b) O recurso “e” pode ser atribuído ao processo P_0 sem risco de *deadlock*? Justifique.