

Número:

Nome:

LEIC/LERC – 2009/10
1º Teste de Sistemas Operativos

14 de Novembro de 2009

Responda no enunciado, apenas no espaço fornecido. Identifique todas as folhas.

Duração: 1h30m

Grupo I [4,1 v.]

1. [0,5 v] Relativamente ao tratamento por lotes, indique a afirmação que está errada (resposta errada desconta $\frac{1}{4}$ da cotação):
- a. É um objectivo maximizar o tempo em que o processador está a correr programas de processos.
 - b. É um objectivo dar a cada processo uma fatia equitativa do tempo.
 - c. É um objectivo evitar a perda de tempo no acesso aos periféricos.
 - d. O mecanismo de interrupções permite evitar a espera activa sobre os periféricos.

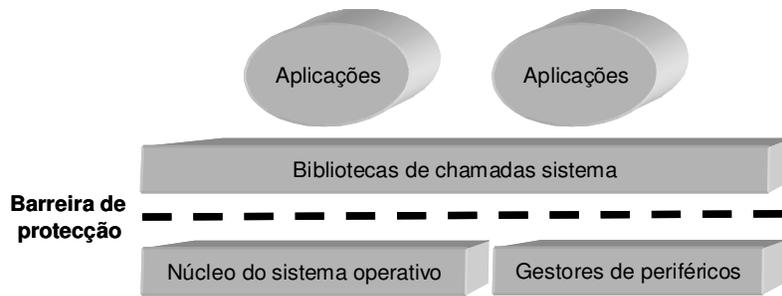
--

2. No computador de bordo de um automóvel, há 2 processos. Existe um processo que está bloqueado à espera do alarme do sistema de travagem do automóvel. Quando desbloqueado, esse processo trata de activar o ABS do carro, que tem de acontecer, no máximo, ao fim de 50ms senão o sistema não cumpre os requisitos. No mesmo sistema, existe um processo menos importante que adquire dados do automóvel e apresenta as estatísticas no ecrã do condutor.

- a. [0,6 v] Considera este sistema de tempo real ou virtual? Porquê? (Seja preciso na formulação do requisito.)

- b. [0,6 v] Assuma que o gestor de processos não é preemptivo. Tal poderá por em causa o correcto funcionamento do ABS? Justifique com um exemplo.

- c. [0,6 v] Num sistema preemptivo, assumo que pode optar por escalonamento com prioridades dinâmicas ou fixas (assumo que em ambos o processo do ABS começa com prioridade maior que o processo das estatísticas). Ambas as opções permitem assegurar o correcto funcionamento do ABS, ou apenas uma delas? Justifique com um exemplo.



3. A figura ilustra a arquitectura típica de um núcleo monolítico.

- a. [0,6 v] Quais são os dois mecanismos incluídos no hardware que permitem criar a barreira de protecção indicada na figura? O que restringem aos processos que correm em modo utilizador?

- b. [0,6 v] Que mecanismos permitem à biblioteca de chamadas sistema invocar operações no núcleo ultrapassando a barreira de protecção? Justifique.

- c. [0,6 v] Quando termina a execução de operações do núcleo como é que o processo volta ao nível de protecção anterior?

Grupo II [3,9 v]

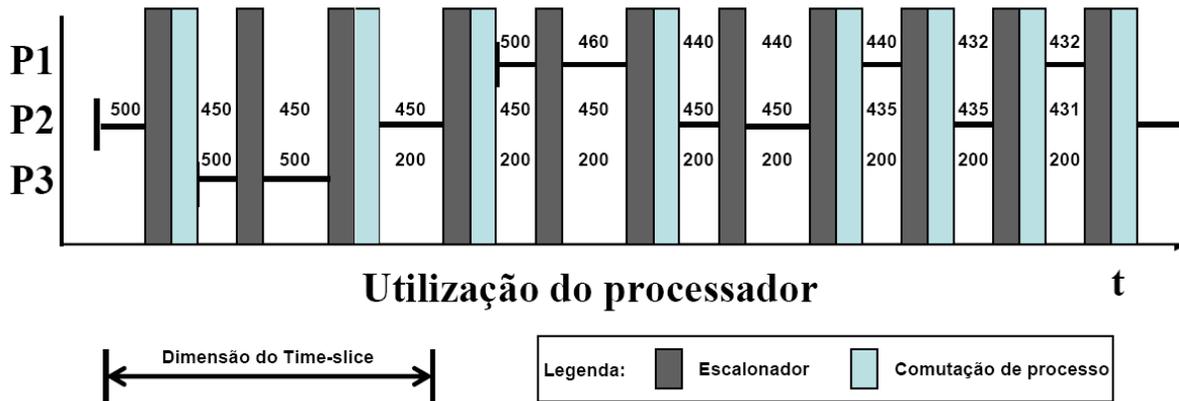
Considere um sistema multi-programado, com um processador único.

O sistema é usado para correr uma aplicação altamente paralela, composta por três processos (P1, P2, P3).

O escalonamento é preemptivo e baseado em prioridades dinâmicas. As prioridades podem variar de 0 a 1000 (sendo a mais alta a mais prioritária). Sobre o cálculo de prioridade, **sabe-se que favorece os processos CPU-intensivos (que, neste caso, são P1 e P2), quando comparados com os processos E/S-intensivos (neste caso, P3):** os processos que se bloqueiam à espera de E/S recebem uma **forte penalização**, enquanto que o uso do CPU é **apenas ligeiramente penalizado** pelo escalonador.

O Time-slice tem a duração indicada na legenda na mesma escala do gráfico. Os valores indicados na figura são as prioridades dos respectivos processos pelo que a evolução do seu valor traduz o escalonamento efectuado.

Os utilizadores da aplicação estão insatisfeitos com o seu desempenho, pelo que se fez uma análise da utilização do processador, apresentada abaixo.



1. Imagine que os programadores da componente de gestão de processos o consultam para pedir a sua opinião sobre possíveis modificações ao escalonador/despacho deste sistema. Indique, para cada modificação, se a considera vantajosa ou desvantajosa, ilustrando porquê com um exemplo da execução acima.

a. [0,7 v.] Inverter o cálculo de prioridade para favorecer tarefas mais intensivas em entradas/saídas (*I/O-intensive*).

b. [0,7 v.] Reduzir a frequência com que o escalonador é executado.

- c. [0,7 v.] Usar escalonador multi-lista apenas com 5 gamas: 0-200, 201-400, 401-600, 601-800, 801-1000.

2. Indique se as seguintes afirmações sobre o escalonamento em Linux são verdadeiras ou falsas, justificando.

- a. [0,6 v.] “Em Linux, é possível a um processo CPU-intensivo estar temporariamente mais prioritário que um processo que se bloqueia frequentemente.”

- b. [0,6 v.] “Quando um processo em execução está em modo núcleo a executar uma função sistema, pode perder o processador devido a três razões: processo terminou, processo bloqueou-se ou outro processo mais prioritário tornou-se executável nesse momento.”

- c. [0,6 v.] “O escalonamento em Linux baseia-se no conceito de época, um período de tempo que é configurado quando o sistema operativo é instalado e que normalmente é de 1s.”

Grupo III [6 v.]

O problema do Supermercado consiste em admitir que existe um supermercado com N caixas de pagamento e com um funcionário em cada caixa.

A tarefa de atendimento consiste no seguinte: enquanto houver clientes na sua fila o funcionário atende-os; se não estiver nenhum cliente para ser atendido na sua fila, o funcionário pode atender um cliente da outra fila; se não existir ninguém para atender em nenhuma das filas, o funcionário bloqueia-se à espera de clientes. A tarefa do cliente consiste no seguinte: quando o cliente chega, espera na fila que tiver menos clientes; uma vez escolhida a fila, o cliente não pode trocar, excepto para ser atendido conforme descrito anteriormente no procedimento da tarefa de atendimento; o número de clientes por fila é ilimitado.

Considere a seguinte solução (com erros) para o problema do supermercado.

Assuma que é executada num processo com tarefas reais que ciclicamente executam EsperarVez, e outras tarefas reais que ciclicamente executam ProximoCliente.

```
/* Funções auxiliares */
// função que retorna a fila com menos clientes
int EscolherFila ();
// função que, caso haja clientes em alguma fila, retorna true e
// indica a fila encontrada em (*filaEncontrada); caso contrário, retorna false
boolean PesquisaFila(int *filaEncontrada);

semaforo_t empregado[N], semFila[N];
boolean empregadoBloqueado[N], emAtendimento[N];
int clientes[N];

void EsperarVez ()
{
    int FilaSel = EscolherFila();
    if (emAtendimento[FilaSel]) {
        clientes[FilaSel]++;
        esperar (semFila [FilaSel]);/* cliente à espera de ser atendido/
        clientes[FilaSel]--;
    }
    if (empregadoBloqueado [FilaSel])
        assinalar (empregado[FilaSel]);
    emAtendimento [FilaSel]= true;
    <Atendimento>
    emAtendimento [FilaSel]= false;
}

void ProximoCliente (int NumEmp)
{
    int FilaSel = numEmp;
    if (emAtendimento[NumEmp]==false || clientes[NumEmp]== 0)
        if ( PesquisaFila(&FilaSel) == False){
            empregadoBloqueado [NumEmp] = true;
            esperar(empregado[NumEmp]);
            empregadoBloqueado [NumEmp] = false;
        };
    if (emAtendimento[FilaSel] == false && clientes[FilaSel] >0)
        assinalar (semFila[FilaSel]);
    <Atendimento>
}

```

1. [0,8 v.] Qual a função do vector de semáforos empregado e a que valor deve ser inicializado?

2. [0,8 v.] O programa está errado porque testa e actualiza variáveis partilhadas fora das secções críticas. Dê um exemplo de erro que poderia suceder. Seja totalmente claro na apresentação do caso concreto de erro.

3. [0,6 v.] Para implementar a secção crítica poderia usar a inibição (disable) das interrupções no início da secção crítica e desinibição no fim da secção crítica? Justifique.

4. [0,8 v.] Para implementar a secção crítica, poderia ter usado um trinco implementado com uma instrução de TestAndSet (que seria fechado no início da secção crítica e aberto no fim da mesma). Apresente uma vantagem e uma desvantagem comparativamente à solução de inibir interrupções.

5. [1 v.] Existe alguma possibilidade de interbloqueagem quando introduzir a solução anterior? Justifique.

6. [1 v.] Corrija a função `Proximocliente`, utilizando trincos lógicos (mutexes).

--

7. [1 v.] Assuma uma solução que assegura exclusão mútua nas secções críticas. Se um cliente fizer Assinalar sobre o semáforo empregado numa altura em que o Empregado não está bloqueado, sucede algum erro? Justifique.

Grupo IV [6 v]

1. Considere uma arquitectura elementar de gestão de memória paginada, com endereçamento virtual de 24 bits, em que cada endereço é dividido em:
12 bits de número de página | 12 bits de deslocamento.

Considere também um processo, P1, cuja tabela de páginas tem o seguinte conteúdo:

Página	Presente	Protecção	Base
0	0	RW	0x001
1	1	RW	0x001
2	0	R	0x001
3	1	RW	0x002
4	0	RW	0x000
5	1	E	0x000

- 1.1. [1 v.] Assuma que P1 faz os seguintes acessos às seguintes posições de memória. Para cada acesso, indique o respectivo endereço real. Caso não disponha de dados suficientes para calcular algum(ns) desse(s) valor(es), responda "Indefinido".

- a. Escrita da posição 0x001F80:
b. Leitura da posição 0x003401:
c. Leitura da posição 0x001F84:
d. Escrita na posição 0x004781:
e. Leitura na posição 0x007381:

- 1.2. Considere os acessos **a**, **c** e **d** apenas (por esta ordem), sabendo que todos estes acessos foram completados com sucesso.

- a. [0,5 v.] Assumindo que P1 corre sozinho, ordene os acessos pelo tempo que prevê que demorem. Assuma que a TLB começa vazias.

--

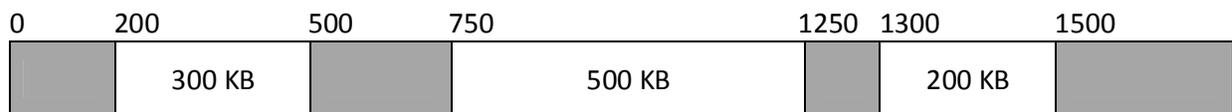
- b. [0,6 v.] Justifique a resposta anterior, indicando, para cada acesso, as componentes (e a respectiva ordem) que foram acedidas durante o acesso à posição de memória. Considere as seguintes componentes: **TLB, tabela de páginas de P1, memória primária, memória secundária.**

Acesso (a):

Acesso (c):

Acesso (d):

2. Considere agora uma arquitectura baseada em segmentos. Num dado momento, a memória física da máquina está ocupada da seguinte forma (a cinzento, segmentos ocupados):



A partir deste estado, assuma que o gestor de memória recebe quatro pedidos de reserva de novos segmentos, na seguinte sequência:

- | |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ol style="list-style-type: none">1. Reserva de novo segmento de 220 Kbytes2. Reserva de novo segmento de 100 Kbytes3. Reserva de novo segmento de 400 Kbytes4. Reserva de novo segmento de 260 Kbytes |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Considere 3 algoritmos diferentes para a reserva dos novos segmentos: Best-fit, Worst-fit e Next-Fit.

- 2.1. Para cada algoritmo, indique: em que posição cada novo segmento é reservado (ou seja, qual a sua base); ou ERRO no caso do algoritmo não encontrar nenhum espaço livre para a reserva. Como sabe, cada algoritmo mantém uma lista ordenada de segmentos livres. Para cada pedido de reserva, apresente também o conteúdo dessa lista após a reserva ter sido feita.

a. [0,6 v.] Algoritmo Best-fit

Estado inicial		Lista de segmentos livres inicial		
		Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 220KB		Lista de segmentos livres resultante		
Resultado (base/ERRO)	<input type="text"/>	Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 100KB		Lista de segmentos livres resultante		
Resultado (base/ERRO)	<input type="text"/>	Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 400KB		Lista de segmentos livres resultante		
Resultado (base/ERRO)	<input type="text"/>	Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 260KB		Lista de segmentos livres resultante		
Resultado (base/ERRO)	<input type="text"/>	Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:

b. [0,6 v.] Algoritmo Worst-fit

Estado inicial		Lista de segmentos livres inicial		
		Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 220KB		Lista de segmentos livres resultante		
Resultado (base/ERRO)	<input type="text"/>	Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 100KB		Lista de segmentos livres resultante		
Resultado (base/ERRO)	<input type="text"/>	Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 400KB		Lista de segmentos livres resultante		
Resultado (base/ERRO)	<input type="text"/>	Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 260KB		Lista de segmentos livres resultante		
Resultado (base/ERRO)	<input type="text"/>	Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:

c. [0,6 v.] Algoritmo Next-fit

Estado inicial		Lista de segmentos livres inicial		
		Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 220KB	Resultado (base/ERRO)	Lista de segmentos livres resultante		
		Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 100KB	Resultado (base/ERRO)	Lista de segmentos livres resultante		
		Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 400KB	Resultado (base/ERRO)	Lista de segmentos livres resultante		
		Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:
Reserva de 260KB	Resultado (base/ERRO)	Lista de segmentos livres resultante		
		Base:	Base:	Base:
		Dim.:	Dim.:	Dim.:

2.2. [0,7 v.] A execução da alínea anterior ilustra bem uma desvantagem do algoritmo Best-fit e uma desvantagem do algoritmo Worst-fit. Indique que desvantagens são essas, e ilustre-as indicando porções da sua resposta à alínea anterior.

2.3. [0,7 v.] Para cada algoritmo, indique o total de fragmentação interna e externa que gerou.

	Fragmentação interna	Fragmentação externa
Best-fit		
Worst-fit		
Next-fit		

2.4. [0,7 v.] Os algoritmos best-fit, worst-fit, next-fit e buddy são também usados com arquiteturas paginadas? Se sim, indique as alterações necessárias aos algoritmos. Se não, indique porquê.
