



Universidade de Aveiro

Departamento de Electrónica, Telecomunicações e Informática

Sistemas de Operação

Exame RE

(Ano Letivo de 2016/17)

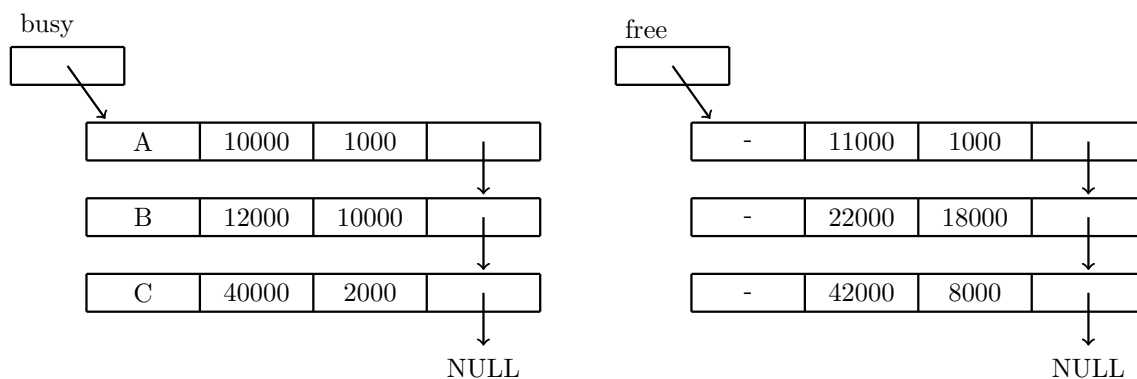
7 de fevereiro de 2017

Nome: _____ NMec: _____

NOTA: Numa questão em que se peça uma justificação e ela não seja dada, a resposta não será considerada.

.....

1. Considere um sistema computacional usando uma organização de memória real de partições variáveis. Como a memória é reservada dinamicamente o sistema de operação tem de manter um registo das regiões de memória (partições) ocupadas e das livres. Isto é feito através de duas listas ligadas. A figura seguinte ilustra os valores dessas listas num dado momento. Cada elemento da lista tem 4 campos, representando a identificação do processo, o endereço inicial da partição, a sua dimensão e um ponteiro para o próximo elemento. A região abaixo do endereço 10000 é reservada para o sistema de operação.



- (a) Compare as organizações de memória real e virtual.
- (b) Numa organização de memória real de partições variáveis, quando um processo é transferido para a memória principal é necessário arranjar uma partição para nela alojar o espaço de endereçamento do processo. *First fit* e *best fit* são duas políticas diferentes usadas para esse fim. Descreva estas políticas.
- (c) Considere que um processo D, reclamando 4000 unidades de memória, vai ser transferido para memória. Apresente os valores das listas **busy** e **free** após a transferência, considerando as políticas *first fit* e *best fit*
- (d) Considere que, na situação ilustrada, o processo B termina, libertando a memória por si utilizada. Apresente os valores das listas **busy** e **free** após a transferência.

2. Considere o programa apresentado a seguir, onde `bwDelay()` é uma função que gera um atraso com tempo aleatório em *busy waiting* (ou seja, não bloqueante).

```
1 void *thread(void *arg)
2 {
3     int i = *((int*)arg);
4     bwDelay();
5     printf("thr %d\n", i);
6     pthread_exit(NULL);
7 }
8
9 int main(int argc, char *argv[])
10 {
11     printf("msg 0\n");
12
13     pthread_t thr[2];
14     int arg[2] = { 0, 1 };
15     int i;
16     for (i = 0; i < 2; i++) {
17         pthread_create(&thr[i], NULL, thread, &arg[i]);
18     }
19
20     bwDelay();
21     printf("msg 1\n");
22
23     for (i = 0; i < 2; i++) {
24         pthread_join(thr[i], NULL);
25     }
26
27     printf("msg 2\n");
28     return 0;
29 }
```

- (a) Defina processo e *thread*, realçando a(s) diferença(s) entre ambos.
- (b) A execução do programa anterior pode resultar na impressão no *standard output* da mensagem.

msg 0; msg 1; thr 0; thr 1; msg 2.

Considerando que a execução de um `printf` é atômica, quantas outras saídas são possíveis. Apresente 3 dessas possíveis saídas. Justifique sucinta e adequadamente a sua resposta.

- (c) Ao preparar o exame, numa versão anterior do programa, o lançamento da *thread* era feita usando

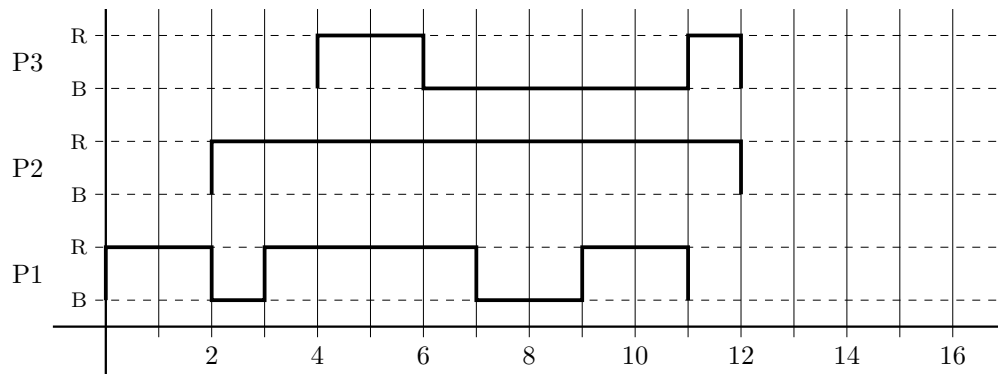
`pthread_create(&thr[i], NULL, thread, &i);`

A execução do programa resultou na saída

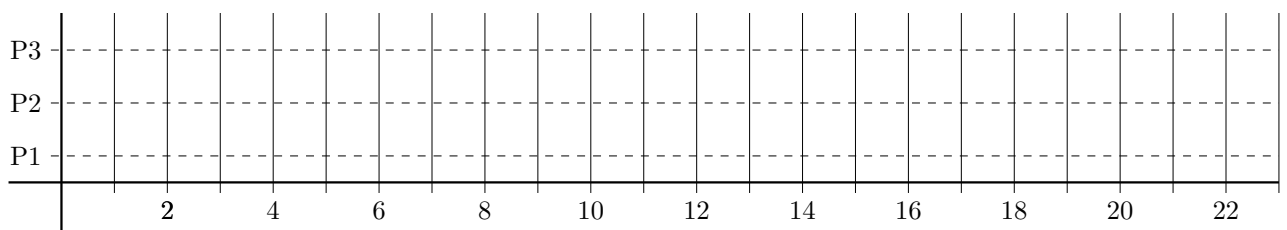
msg 0; msg 1; thr 0; thr 2; msg 2.

Se, dentro do `for`, a variável `i` assume os valores 0 e 1, como é possível que haja na saída um `thr 2`? Justifique convenientemente a sua resposta.

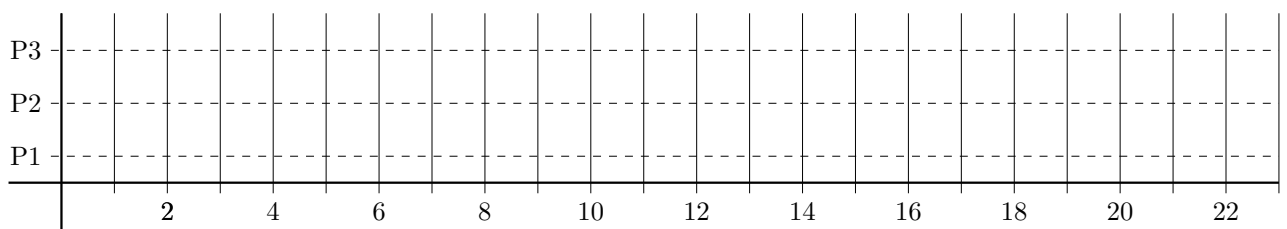
3. O gráfico seguinte representa o estado da execução de 3 processos independentes entre si (mesmo em termos de I/O), P1, P2 e P3, assumindo que correm em processadores (virtuais) distintos. R e B indicam, respetivamente, que o processo está no estado RUN (a usar o processador) ou no estado BLOCKED (bloqueado à espera de um evento).



- (a) Os processos podem ser agrupados em níveis de prioridade distinta no que respeita ao acesso ao processador. Desenhe o diagrama de estados de um escalonador de processador de baixo nível de um sistema multiprogramado com suporte a prioridades. Em que circunstâncias se dá a transição do estado RUN para o estado Ready to Run?
- (b) As prioridades dos processos podem ser de dois tipos: estáticas e dinâmicas. Distinga estes dois tipos de prioridade.
- (c) Considere que os 3 processos representados acima correm num ambiente multiprogramado monoprocesso. Usando o gráfico abaixo, trace o diagrama temporal de escalonamento do processador pelos processos P1, P2 e P3, considerando uma política de escalonamento Round Robin sem prioridades e com um *time quantum* (*time slot* atribuído a cada processo) de 3.



- (d) Considere que os 3 processos representados acima correm num ambiente multiprogramado monoprocesso. Usando o gráfico abaixo, trace o diagrama temporal de escalonamento do processador pelos processos P1, P2 e P3, considerando uma política de escalonamento FCFS (First Come First Served), em que o processo P3 tem maior prioridade que os outros dois, que têm a mesma prioridade.



4. Considere que 3 processos (P1, P2 e P3) partilham recursos de 3 categorias diferentes (R1, R2 e R3). Para executarem até ao fim, os processos necessitam de reter simultaneamente um certo número de recursos de cada tipo. As tabelas seguintes ilustram os recursos disponíveis e os estados dos processos em termos de recursos já adquiridos e dos recursos ainda por adquirir.

Recursos disponíveis			Estados dos processos						
			Recursos já adquiridos			Recursos por adquirir			
R1	R2	R3		R1	R2	R3	R1	R2	R3
0	2	3	P1	3	1	4	2	0	0
			P2	2	1	0	0	2	1
			P3	2	1	0	3	1	0

- (a) Distinga políticas de prevenção de deadlock em sentido estrito (*deadlock prevention*) e em sentido lato (*deadlock avoidance*).
- (b) Mostre que existe uma ordem de execução dos processos que permite que todos terminem, sem entrar em deadlock.
- (c) Se o processo P3 pede e obtém um recurso do tipo R2, o sistema pode entrar em deadlock? Justifique a sua resposta.
5. A interação entre um sistema computacional e os dispositivos de I/O pode ser feita por *Polled I/O*, *Interrupt driven I/O* e *DMA based I/O*.
- (a) Qual escolheria para ligar um teclado ao sistema computacional? Justifique a sua resposta.
- (b) Qual escolheria para ligar um disco magnético ao sistema computacional? Justifique a sua resposta.