UNIVERSIDADE DE AVEIRO

DEPARTAMENTO DE ELECTRÓNICA, TELECOMUNICAÇÕES E INFORMÁTICA

Teste de Sistemas Operativos

10/Janeiro/2017

Nome: _____ Nº mec. _____

NOTE BEM: Justifique todos os passos das suas respostas. Respostas não justificadas não serão consideradas.

I. Considere o seguinte código:

```
1.  int main(int argc, char *argv[])
2.  {
3.    switch(fork()) {
4.      case 0:  printf("A\n");
5.               break;
6.      default: printf("B\n");
7.               wait(NULL);
8.               printf("C\n");
9.    }
10.   return 0;
11. }
```

a) Indique quais as linhas do código anterior que são executadas pelo processo pai e quais as linhas que são executadas pelo processo filho.

b) Apresente duas saídas com conteúdo distinto que podem resultar da execução do código anterior.

c) Diga por palavras suas o que entende por processo e *thread*, indicando claramente o que distingue um processo de uma *thread*.

d) Desenhe o diagrama de estados de um processo.

e) Considerando o diagrama de estado de um processo, indique, para cada linha de código que o processo pai executou, qual o estado (ou estados) em que esse processo poderia estar.

II. Considere que o código seguinte é executado por 5 *threads*, representando cada *thread* um filósofo do problema do Jantar dos 5 Filósofos. Cada filósofo é identificado pelo parâmetro **`f`** que varia entre 0 e 4. As *threads* têm acesso a um *array* de semáforos **`forks`** com 5 semáforos (indíce de 0 a 4) que representam os garfos.

```
void filosofo(int f) {
  while(true) {
    think();
    //getForks
    forks[left(f)].down();
    forks[right(f)].down();
    eat();
    //putForks()
    forks[left(f)].up();
    forks[right(f)].up();
  }
}
```

a) Implemente as funções **`int left(int f)`** e **`int right(int f)`** que retornam para o filósofo **`f`**, o identificador do garfo à sua esquerda e à sua direita, respectivamente.

b) Mostre que o código anterior pode conduzir a situações de *deadlock*.

c) Altere o código anterior, sem adicionar recursos de sincronização, de modo a evitar a ocorrência de *deadlock*.

III. Considere que 3 processos (P1, P2 e P3) partilham 3 recursos (R1, R2, R3). Para executarem até ao fim, os processos necessitam de deter simultaneamente um certo número de recursos. A situação é descrita pelas tabelas seguintes:

Recursos disponíveis:

R1	R2	R3
0	2	3

Estado dos processos:

	Recursos já adquiridos			Recursos a pedir		
	R1	**R2**	**R3**	**R1**	**R2**	**R3**
P1	3	1	4	2	0	0
P2	2	1	0	0	2	1
P3	2	1	0	3	1	0

a) Podem os processos P1, P2 e P3 terminar? Justifique a sua resposta, apresentando, no caso de ser possível, uma sequência de execução dos processos P1, P2 e P3 que permita que todos terminem.

b) Considere que o processo P3 pede, com sucesso, mais um recurso do tipo R2. Caracterize a nova situação criada por este pedido quanto à ocorrência de *deadlock*? Justifique.

c) Considere que os processos P1, P2 e P3 partilham 1 semáforos (**`s_mutex`**), que foi inicializado com o valor 1, e que tem disponíveis os métodos **`.up()`** e **`.down()`**. Os processos partilham também a estrutura de dados:

```
struct inf {  int avail_r1, avail_r2, avail_r3; } info;
```

Escreva a rotina **`void print_avail_resources(void)`**, que, em exclusão mútua, escreve na consola o número de recursos disponíveis de cada tipo.

IV. Considere que os processos P1, P2 e P3 têm um tempo total de execução de 100ms, 50ms e 10ms.

a) Apresente o diagrama temporal do escalonamento deste processos usando o algoritmo *Round Robin* com um *quantum time* de 20ms.

b) Qual o tempo médio de espera dos processos para o escalonamento da alínea anterior?

c) Apresente as vantagens e desvantagens do escalonador *Round Robin* relativamente ao escalonador *FCFS*. Indique também, justificando, qual o mais indicado para escalonar aplicações interactivas e qual o mais indicado para escalonar aplicações de cálculo intensivo.

V. O conteúdo da memória e a tabela de página de cada processo activo de um sistema de memória virtual paginada são apresentados a seguir (de forma simplificada, considerando que cada página tem o tamanho de 1 byte e que apenas existem 2 processos):

P1

	Memória	Tabela de página
0:	**A**	**5**
1:	**B**	**1**
2:	**C**	**3**
3:	**D**	**8**

P2

	Memória	Tabela de página
0:	**E**	**9**
1:	**F**	**4**
2:	**G**	**0**
3:	**H**	**11**

a) Apresente o conteúdo da memória física entre os endereços 0 e 15.

b) Pode um processo alterar directamente o conteúdo da sua tabela de página? Justifique.

c) Quais os principais objectivos da Memória Virtual?