MIEIC SISTEMAS OPERATIVOS – 2009/2010 1S FEUP

Tipo de prova: sem consulta **Exame Normal da Época Normal**
Duração: 2 horas **25 de Janeiro de 2010**
Cotação máxima: 20 valores
Estrutura da prova: Parte I (escolha múltipla, 25%); Parte II (teórica, 50%); Parte III (Aplicação, 25%)

Nota: Utilize folhas de resposta independentes para cada uma das partes.

Parte I: Escolha Múltipla [5 valores]

Utilização: para cada pergunta só há uma resposta correcta; indique-a com a letra correspondente na folha de respostas; se não souber a resposta correcta, não preencha ou faça um traço nessa alínea.

Cotação: cada resposta certa vale 1 ponto; cada resposta errada vale –0.3 pontos; cada resposta ambígua, ininteligível ou não assinalada vale 0 ponto. O total é 15 pontos.

1. Quais das seguintes instruções só pode ser executada em modo de supervisão (*kernel mode*)?
   a. Desactivação de interrupções
   b. Leitura do relógio interno
   c. Alternar entre modo utilizador (*user mode*) e de supervisão

2. Um computador com um único CPU contém 512MB de memória RAM, dos quais 128MB são ocupados pelo sistema operativo. Cada processo ocupa 128MB. Qual é a percentagem de utilização do CPU se os processos tiverem um tempo médio de entrada/saída (*I/O wait*) de 80%?
   a. 39%
   b. 49%
   c. 59%

3. No mesmo computador da pergunta 2., como poderia chegar a um tempo médio de utilização do CPU de 91%?
   a. Adicionando 1024MB de memória
   b. Adicionando 512MB de memória
   c. Adicionando 128MB de memória

4. Um computador contém 4 *page frames*. O tempo para carregar, hora do último acesso, os bits R e M das páginas estão na tabela (tempos estão em *clock ticks*). Que página será substituída usando o algoritmo NRU (*not recently used*)?

Page	Loaded	Last ref.	R	M
0	126	280	1	0
1	230	265	0	1
2	140	270	0	0
3	110	285	1	1

   a. 1
   b. 2
   c. 3

5. Um programa não deverá ter mais do que uma zona de memória crítica
    a. Verdadeiro
    b. Falso, mas deverá ser o menor número possível
    c. Somente verdadeiro em sistemas UNIX

6. Indique qual dos seguintes métodos não permite enviar o sinal SIGINT a um processo
    a. `ps -kill`
    b. `int kill(pid_t pid, int sig)`
    c. `sigalarm`

7. Qual dos seguintes não é uma camada do sistema operativo?
    a. Kernel
    b. Programa de aplicação
    c. Secção crítica

8. Qual o output produzido pelo seguinte comando "echo $PWD | cat"?
    a. Imprime o valor da variável $PWD e executa `cat`
    b. O output produzido é igual a "echo $PWD & cat" e "cat < echo $PWD"
    c. O comando `cat` envia para o *stdout* o valor da variável $PWD pois o output do comando `echo` é redireccionado para o input do comando `cat`

9. Um algoritmo de escalonamento determina a ordem de execução dos processos. Se 3 processos estiverem prontos a executar, quantas maneiras diferentes de execução existem (assuma (i) que todos os processos terminam e (ii) não preempção)?
    a. 3
    b. 6
    c. 9

10. *Thrashing*
    a. Reduz a *page I/O*
    b. Diminui o grau de multiprogramação
    c. Implica excessivo *page I/O*

11. Semáforos são oferecidos pelo sistema operativo para ajudar na resolução de
    a. Wait & Signal
    b. Deadlocks
    c. Sincronização

12. Em sistemas interactivos, o requisito essencial é ter um bom tempo de resposta e dividir os recursos de forma igual. Qual o melhor algoritmo de escalonamento?
    a. *Shortest remaining time next*
    b. *Priority preemptive scheduling*
    c. Round-Robin

13. Demand page memory allocation
    a. Permite que os endereços virtuais sejam independentes dos endereços físicos
    b. Permite que os endereços virtuais sejam múltiplos dos endereços físicos
    c. Só está presente no sistema operativo Windows NT

14. Qual dos seguintes métodos usado pelo sistema operativo deve ser do conhecimento do programador de aplicações
    a. Paginação
    b. Segmentação
    c. Swapping

15. Dos seguintes nomes de ficheiros, qual deles é um nome absoluto
    a. /usr/bin/calendar
    b. ./calendar
    c. ../usr/bin/calendar

Parte II: Predominantemente teórica [10 valores]

Utilização: respostas devem ser (brevemente) justificadas.

Cotação: indicada em cada pergunta.

1. **[1 val.]** Quais são as duas funções principais dum sistema operativo? Justifique contextualizando historicamente o aparecimento dos sistemas operativos.

2. **[1 val.]** Quais as diferenças fundamentais entre processos e threads (indique pelo menos 3 aspectos que distingam estes dois conceitos)?

3. **[1 val.]** Indique algumas diferenças (pelo menos duas) entre computadores pessoais e mainframes.

4. **[2 val.]** Qual a diferença, em termos de *stdout*, entre os dois pseudo-programas em baixo? Fundamente a sua resposta, indicando qual o processo, de entre o pai e filho, que considera que executa primeiro.

Pseudo-programa 1:

```
if(fork() == 0) {
        execlp("who", "who", NULL);
        print("a terminar filho\n");
} else {
        print("a terminar pai\n");
}
```

Pseudo-programa 2:

```
if(fork() == 0) {
        system("who", "who", NULL);
        print("a terminar filho\n");
} else {
        print("a terminar pai\n");
}
```

5. **[1 val.]** Quantas *page faults* irão ocorrer na string de referência 0172327103 num sistema com quatro *page frames* e 8 *pages* se o algoritmo de substituição for o FIFO e as *page frames* estiverem inicialmente vazias? E com o LRU? Justifique a sua resposta.

6. **[1 val.]** O que é uma *trap*? Qual a diferença entre *traps* e *interrupts*?

7. **[1 val.]** Distinga em dois aspectos a utilização de canais normais e canais com nome. Quais as principais diferenças entre um canal com nome e um ficheiro?

8. **[2 val.]** Suponha uma situação de *deadlock* envolvendo três processos
    a. Represente-a esquematicamente usando um grafo.
    b. Enumere duas razões para o surgimento do *deadlock*.
    c. Estarão os processos necessariamente no estado bloqueado?
    d. Como poderia ser o problema evitado (dica: algoritmo de *deadlock avoidance*)?
    e. E detectado (dica: detecção de *deadlocks* com matrizes)?

Parte III: Predominantemente de aplicação [5 valores]

Utilização: respostas devem ser (brevemente) justificadas.

Cotação: indicada em cada pergunta.

1. **[2.5 val.]** Considere o código fonte do programa `xpto` em baixo. Como pode verificar, este programa deve ser invocado com dois argumentos identificando dois *named pipes*. Por uma questão de simplificação, omitiu-se o controlo de erros e assumiu-se que os *named pipes* já estão criados.

```
1.  #include<stdio.h>
2.  #include<signal.h>
3.  void recebe(char *de) {
4.          char ch;
5.          int fd;
6.          fd = open(de, O_RDONLY);
7.          while(read(fd, &ch, 1)) {
8.                  write(1, &ch, 1);
9.          }
10.         close(fd);
11. }
12. void envia(char *para) {
13.         char ch;
14.         int fd;
15.         fd = open(para, O_WRONLY);
16.         while(read(0, &ch, 1)) {
17.                 write(fd, &ch, 1);
18.         }
19.         close(fd);
20. }
21. int main(int argc, char *argv[]) {
22.         int status;
23.         int p;
24.         if(p = fork()) {
25.                 recebe(argv[1]);
26.                 //wait(&status);
27.                 //kill(p, SIGKILL);
28.         } else {
29.                 envia(argv[2]);
30.                 //kill(getppid(), SIGKILL);
31. }
```

   a. Analise cuidadosamente o código e diga, de forma sucinta, qual é a funcionalidade implementada pelo mesmo. Centralize a sua resposta em torno do que é lido/escrito dos/nos quatro descritores usados (*stdin*, *stdout* e os dois descritores associados ao *argv[1]* e *argv[2]*).
   b. Suponha que o programa é invocado em paralelo, em duas shells, da seguinte forma:

      shell1 $>./xpto ana2rita rita2ana e *shell2 $>./xpto rita2ana ana2rita*

      i. Assumindo que **ana2rita** e **rita2ana** são dois *named pipes* previamente criadas, diga o que acontece quando se introduz na shell2, via teclado, "Ola Ana".
      ii. Quando se acciona Ctrl+C numa das shells os 4 processos terminam todos? Justifique.
   c. Na sua opinião, qual (ou quais) das 3 linhas comentadas deve ser

descomentada? Justifique sucintamente a sua opção em termos do impacto para o utilizador e para o sistema. Teria a mesma opção se soubesse que o programa seria maioritariamente invocado com o *stdin* redireccionado para um ficheiro de dimensão considerável (ex., ./xpto pipe1 pipe2 < /var/log/messages)?

d. Na sua opinião, o uso do *fork()* é mesmo necessário? Não seria suficiente invocar a função *recebe* seguida da função *envia*? Justifique.
e. Suponha que o programa era invocado com dois parâmetros, identificando nomes de ficheiros em vez de *named pipes.*
    i. Qual seria o output do programa na consola?
    ii. Qual seria o conteúdo do segundo ficheiro após o término do programa, se a linha 27 estiver descomentada e o primeiro ficheiro estiver vazio ou for muito pequeno?

2. [2.5 val.] Considere o código em baixo. Por uma questão de simplificação e espaço disponível omitiu-se o código para controlo de erros.

```
1.  #include<stdio.h>
2.  #include<pthread.h>
3.  #define MAX 5
4.  #define NT 10
5.
6.  int produtos[MAX];
7.  void* recebe(voip* p) {
8.          static int r = 0;
9.          while(1) {
10.             r %= MAX;
11.             printf("Recebi o %d\n", produtos[r++]);
12.             sleep(1);
13.         }
14.         return NULL;
15. }
16. void* envia(void* p) {
17.         static int r = 0;
18.         static int prod = 0;
19.         while(1) {
20.             r %= MAX;
21.             printf("%d Enviei o %d\n", (int)p, produtos[r++] = ++prod);
22.             sleep(5*(int)p);
23.         }
24.         return NULL;
25. }
26. int main(int argc, char *argv[]) {
27.         pthread_t tenvio[NT], trecebe;
28.         int i;
29.         for(i = NT; i > 0; ) {
30.             pthread_create(&tenvio[i], NULL, envia, (void*)i--);
31.         }
32.         pthread_create(&trecebe, NULL, recebe, NULL);
33. }
```

a. O programa termina inesperadamente, após imprimir algumas linhas. Porquê? Seja sucinto na sua explicação.
b. Sugira uma linha de código adicional para evitar o comportamento descrito na alínea anterior.
c. Na sua opinião, as funções recebe e envia precisam de trincos? Se sim, onde os colocaria?
d. Indique onde, como e que semáforos colocaria para evitar que:
    i. Haja produtos que nunca são recebidos.
    ii. Haja produtos que são recebidos em duplicado.

RMA, JVV, HSF