Número: Nome:

LEIC/LERC – 2010/11
1º Exame de Sistemas Operativos

12 de Janeiro de 2011
Responda no enunciado, apenas no espaço fornecido. Identifique todas as folhas.
Duração: 2h30m

Grupo I [5 valores]

Um conjunto de empregados trabalha numa pequena empresa de execução de obras de construção civil, onde os recursos são limitados. A empresa efectua vários tipos de obras desde canalização, a pintura, colocação de chão e de tecto. Para cada uma dessas tarefas existem ferramentas específicas que são partilhadas por todos os empregados. Quando é atribuída uma tarefa a um empregado ele consulta uma tabela com as ferramentas que vai necessitar para a executar.

Existem 5 tipos de ferramentas diferentes. Um empregado pode necessitar de zero ou mais ferramentas de cada tipo, i.e. necessita apenas de alguns tipos de ferramenta e pode necessitar de mais do que um elemento de um tipo. Estão tabeladas 10 tarefas diferentes.

Assuma a existência de duas tabelas int ferramentas[10][5] e int ferramentasDisp[5] com o tipo de ferramentas por cada tarefa e o Nº de ferramentas disponível inicialmente, respectivamente. Por exemplo se a tabela int ferramentasDisp = {8,5,7,1,100} isso quer dizer que existem no máximo 8 ferramentas do tipo 0, 5 ferramentas do tipo 1, 7 ferramentas do tipo 2, etc. Por outro lado se int ferramentas[0]= {1,2,3,4,5} isso quer dizer que para executar a tarefa 0 são necessárias, 1 ferramenta do tipo 0, 2 do tipo 1, etc.

Quando um empregado quer efectuar o trabalho executa a função "void queroRealizarTarefa(int id, int tipo_tarefa)". Caso não encontre de imediato todas as ferramentas de que necessita, deve ficar bloqueado.

Quando acabar a tarefa deve executar a função "void acabeiTarefa(int id, int tipo_tarefa)".

O código seguinte utiliza um semáforo de unidades por tipo de ferramenta para sincronizar a aquisição de ferramentas por parte dos vários candidatos. Note que um semáforo de unidades possui as funções de "esperar" e "assinalar" comuns, mas onde são recebidas e enviadas mais do que uma unidade em simultâneo:

```
void esperar(semaphore_t semáforo, int unidades);
void assinalar(semaphore_t semáforo, int unidades);
semaphore_t createSem(int uInicias); // cria um semáforo e inicia-o com uIniciais unidades
```

```
semaphore_t sem[5];
void init() {
     for(int i=0;i< 5;i++) sem[i] = createSem(ferramentasDisp[i])
}

void queroRealizarTarefa(int id, int tipo_tarefa) {
     for(i=0;i<5;i++)
        if(ferramentas[tipo_tarefa][i]>0)
          esperar(sem[i],ferramentas[tipo_tarefa][i]);
```

```
}

void acabeiTarefa(int id, int tipo_tarefa) {
     for(i=0;i<5;i++)
        if(ferramentas[tipo_tarefa][i]>0)
          assinalar(sem[i],ferramentas[tipo_tarefa][i]);
}
```

1) [1v] Se as operações de esperar e assinalar fossem substituídas pelas operações:
   a. ferramentasDisp [i]-ferramentas[tipo_tarefa][i]
   b. ferramentasDisp [i]+ferramentas[tipo_tarefa][i]

   O que poderia acontecer? Ilustre com um exemplo.

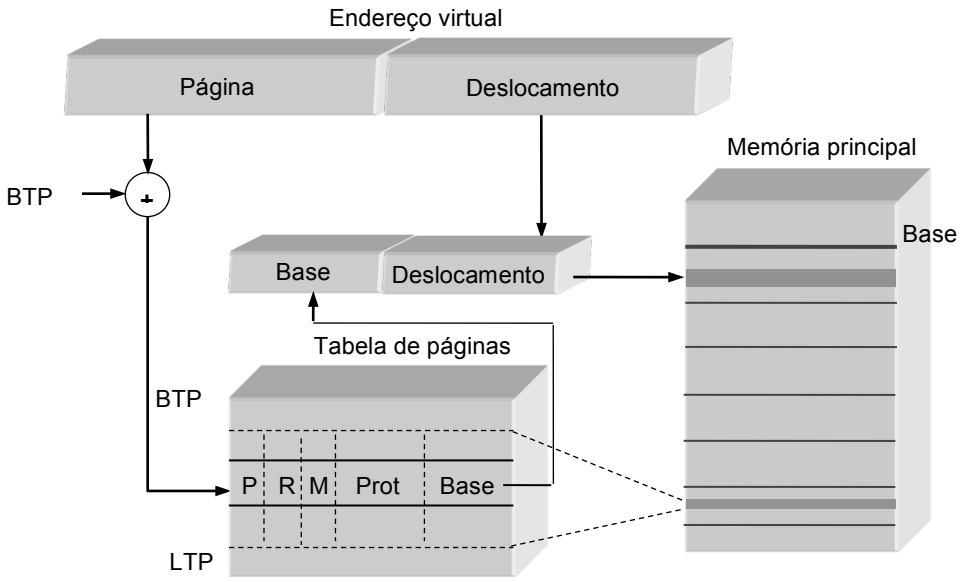
2) [1,5v] Assumindo agora que não possui semáforos de unidades mas simples semáforos com as operações de esperar e assinalar. Modifique o código inicial (não o alterado pelas alíneas 1 e 3) de modo a obter a mesma funcionalidade. Sugestão: Utilize e manipule directamente o vector ferramentasDisp.

3) [1v] Na versão apresentada os trabalhadores recolhem a sua ferramenta pela ordem do seu identificador. Alternativamente, poderiam adquirir a ferramenta pela ordem da que mais necessitam. Assim no caso de ferramentas[0]= {1,2,3,4,5} o trabalhador para executar a tarefa 0, começaria por adquirir 5 unidades da ferramenta 4, depois 4 da ferramenta 3 e

assim sucessivamente. Uma das estratégias está errada. Diga qual, a agora descrita ou a original. Porquê?

4) [1,5v] A solução apresentada limita fortemente a concorrência. Apresente uma solução usando **monitores** com um maior grau de concorrência (admitem-se soluções que sofram de míngua)..

Grupo II [5 valores]



Considere a figura apresentada acima no âmbito da memória virtual paginada.

1) [1v] Complete a figura (desenhando directamente na figura) de modo a que a tradução de endereços seja optimizada. Justifique a sua resposta explicando o funcionamento da optimização que indicou.

2) [1v] Diga qual a utilidade de cada um dos elementos da PTE.

3) [1v] Imagine um computador no qual as páginas virtuais têm uma dimensão de 8192 bytes cada e que o espaço de endereçamento virtual é 4 Gigabytes. Diga como é que é constituído um endereço virtual.

4) [1v] Diga qual o efeito de um "*context switching*" nas estruturas indicadas na figura. Tenha em conta a sua resposta à primeira questão deste grupo.

5) [1v] Considere a noção de working-set de um processo. Diga o que entende por esta noção. Qual a implicação desta noção ao nível da paginação efectuada pelo sistema operativo?

Grupo III [2,7 valores]

1) Em Unix, uma vez aberto um ficheiro, ele é identificado por um inteiro (no âmbito do processo que o abriu).

```
int fd = open("/home/root/fileA", O_READONLY);
```

Consultando diversas estruturas de dados, a partir desse inteiro (fd no exemplo) consegue-se determinar o nº dos blocos em disco que armazenam o conteúdo do ficheiro.

a. [0,5v] Em vez de obrigar o programador a abrir um ficheiro sempre que o queira usar, poder-se-ia pensar numa API alternativa em que as funções *read* e *write* recebiam o caminho de acesso do ficheiro como argumento, não precisando o programador de chamar *open* previamente. Do ponto de vista do desempenho, porque é esta variante desvantajosa?

b. [0,5v] Indique o nome das diversas estruturas de dados que, se consultadas, permitem traduzir o inteiro *fd* nos números dos blocos do ficheiro. Apresente-as pelo ordem por que são consultadas.

c. [0,5v] Imagine que, num outro sistema operativo, o inteiro *fd* consistia no nº do inode do ficheiro. Sempre que o processo solicitasse uma leitura ou escrita ao ficheiro, *fd* era directamente usado para ler o inode no disco, a partir do qual se indexavam os blocos do ficheiro. Indique duas desvantagens desta solução, relativamente à solução usada em UNIX.

2) Considere a seguinte estrutura de um inode:

```
typedef struct fs_inode {
    fs_itype_t   type;
    unsigned int size;
    unsigned int blocks[INODE_NUM_BLKS];
} fs_inode_t;
```

a. [0,3v] Para que serve o campo *type*?

b. Dois sistemas de ficheiros diferentes usam esta estrutura de dados como inode:

- Sistema de ficheiros FS1: usa todos elementos do array blocks como referências directas para blocos, sendo INODE_NUM_BLKS = 2058.
- Sistema de ficheiros FS2: tem INODE_NUM_BLKS = 11, sendo que as primeiras 10 entradas são referências directas para blocos, enquanto que a 11ª entrada é uma referência para um bloco em disco que contêm 2048 entradas directas.

i. [0,3v] Qual sistema de ficheiros é mais rápido no acesso a ficheiros pequenos (até 10 blocos)? (Resposta errada desconta 1/3.)

FS1 FS2 Nenhum, ambos são semelhantes nesse aspecto

ii. [0,3v] Qual sistema de ficheiros é mais rápido no acesso a ficheiros grandes (mais de 10 blocos)? (Resposta errada desconta 1/3.)

FS1 FS2 Nenhum, ambos são semelhantes nesse aspecto

iii. [0,3v] Qual sistema de ficheiros ocupa mais espaço de disco com meta-datos, assumindo que a dimensão média de um ficheiro é 6 blocos? (Resposta errada desconta 1/3.)

FS1 FS2 Nenhum, ambos são semelhantes nesse aspecto

Grupo IV [3,5 valores]

1) [0,5v] Considere que pretende comunicar entre dois processos em máquinas diferentes. Qual das frases **está totalmente correcta? (Resposta errada desconta 1/5 da cotação.)**
   a. Utilizo Sockets em domínio AF_UNIX
   b. Utilizo sockets em domínio AF_INET mas é obrigatório que as máquinas tenham o sistema operativo Unix
   c. Utilizo Sockets no domínio AF_INET sem restrições quanto ao sistema operativo
   d. Utilizo Sockets no domínio AF_INET mas apenas do tipo stream
   e. Utilizo Sockets no domínio AF_INET mas apenas do tipo datagram

2) [0,5v] As frases seguintes comparam a Memória Partilhada e a Comunicação por envio de mensagens. Qual das frases **é correcta? (Resposta errada desconta 1/5 da cotação.)**
   a. A comunicação com memória partilhada é menos complexa de programar porque a sincronização é implícita e o número de chamadas sistema a utilizar é menor.

b. A comunicação por envio de mensagens é mais rápida mas a complexidade de programação da sincronização também é maior.
c. A comunicação por envio de mensagens é menos rápida mas a complexidade de programação da sincronização também é menor.
d. A comunicação por envio de mensagens é mais rápida mas a complexidade de programação da sincronização também é menor.
e. A comunicação por envio de mensagens só é utilizável para um modelo de difusão.

3) Considere o ciclo principal de um servidor que faz apenas o eco das mensagens que lhe chegam de clientes. Os clientes podem interactuar com o servidor quer com ligações quer com datagrams.

```
1 for(;;) {
2   mask = testmask;
3   select(MAXSOCKS,&mask,0,0,0);
4   if(FD_ISSET(strmfd,&mask)) {
5     clilen = sizeof (cliaddr);
6     newfd = accept(strmfd,(struct sockaddr*)&cliaddr, &clilen);
7     echo(newfd);
8     close(newfd);
9   }
10  if(FD_ISSET(dgrmfd,&mask)) echo(dgrmfd);
11 }
```

a. [0,5v] Em que linha o servidor se bloqueia à espera de receber mensagens em stream? Justifique.

b. [0,5v] Explique o funcionamento da variável mask indicando o que valor deve ter na linha 3 antes da invocação do select e que valor ou valores terá quando for testada na linha 4.

c. [0,5v] Quantos sockets devem ter sido criados antes do servidor executar esta secção de código? Justifique.

d. [0,5v] Em que situação é criado mais algum socket na sequência acima? Justifique.

e. [0,5v] Se a chamada sistema da linha 8 não existisse o que poderia suceder? Justifique.

Grupo V [3,8 valores]

1) [0,8v] A maioria dos periféricos em Unix são identificados por dois números, diga quais e o que representa cada um deles.

2) [1v] Considere a seguinte sequência de operações sobre um periférico:

```
int fd=open("/dev/tty0", O_WR);
write(fd, "Ola mundo cruel!",16);
```

Para executar a linha 2, é necessário chamar a função de escrita do gestor de periférico associado ao periférico "/dev/tty0". Para localizar essa função, o sistema operativo consulta várias tabelas e estruturas de dados. Indique quais, e por que ordem.

3) [1v] Em Linux o gestor de periféricos da interface gráfica é efectuada pelo servidor X, indique as vantagens e desvantagens de ter um gestor de periféricos fora do núcleo, quantifique a sua argumentação sempre que possível.

4) [1v] Para além dos gestores de periféricos acedidos a partir do nome no directório /dev e do gestor da interface gráfica, indique que outros gestores existem em Linux e como são identificados e acedidos.