P1 - Sistemas Operacionais I
Professor: Leandro Marzulo
2012-2

Nome:

Instruções: Esta prova é composta de seis questões totalizando 12 (doze) pontos, sendo a nota máxima 10 (dez). Responda as questões de forma sucinta e clara. O uso de lápis é permitido, no entanto, pedidos de revisão serão considerados apenas para questões respondidas a caneta. BOA PROVA!

1) (3,0) Com relação às técnicas de E/S, responda:

a) (1,0) O que é e para que serve um spooling de impressão?
Resp.: No momento em que um comando de impressão é executado, as informações que serão impressas são gravadas antes em um arquivo em disco, conhecido como arquivo de spool, liberando imediatamente o programa para outras atividades. Posteriormente, o sistema operacional encarrega-se em direcionar o conteúdo do arquivo de spool para a impressora.

b) (1,0) Nos primeiros computadores, cada byte em um dispositivo de E/S era lido ou escrito diretamente pelo processador, isto é, não havia nenhum mecanismo para realizar a transferência de bytes diretamente entre o dispositivo e a memória. Que implicações esse arranjo tem para a multiprogramação? Qual o nome da técnica concebida para eliminar este problema?
Resp.: Quando não existir uma controladora de DMA no hardware, o processador será o responsável pelas transferências de dados entre os dispositivos físicos e a memória principal do computador. Com isso, o processador tenderá a estar ocupado na maior parte do tempo, mesmo que os processos executem operações de E/S com frequência. Isso ocorrerá porque para a maior parte dos dispositivos (com exceção dos mais lentos, como, por exemplo, um modem ligado a uma linha telefônica) o tempo da transferência dos dados entre a memória e o dispositivo dominar ao tempo total da operação de E/S. Logo, o principal ganho da multiprogramação, que é o de evitar que o processador fique ocioso quando operações de E/S são executadas, será reduzido, e a multiprogramação essencialmente permitirá a execução de vários processos no processador. O nome da técnica é DMA (Direct Memory Access).

c) (1,0) Poderíamos ter multiprogramação sem interrupções? Justifique.
Resp.: Não, pois é em função desse mecanismo que o sistema operacional sincroniza a execução de todas as suas rotinas e dos programas dos usuários, além de controlar dispositivos.

2) (1,5) Qual a diferença entre processos e threads? Cite as funções usadas no compartilhamento de memória entre processos no linux e explique seu funcionamento.

Resp.: Um processo pode possuir múltiplas threads. Threads representam um linha de execução de um programa e, como os processos, podem ser escalonadas pelo SO (se o mesmo tiver suporte para threads). Threads compartilham o espaço de endereçamento (exceto a pilha de chamadas). Sendo assim, não é necessário criar regiões de memória compartilhada para fazer a comunicação entre threads. A criação de threads é menos custosa pois o espaço de endereçamento do processo não precisa ser replicado.

As funções para compartilhamento de memória entre processos, no linux são:
- **shmget() – SHared Memory GET – Cria a região de memória compartilhada e retorna um identificador para o segmento da região**
- **shmat() – SHared Memory ATtatch – anexa a região criada ao seu espaço de endereçamento do processo que deseja usá-la (precisa do identificador para o segmento)**
- **shmdt() – SHared Memory DeTtatch - Desanexa memória**
- **shmctl() – SHared Memory ConTroL - Operações de controle sobre a região compartilhada - é usada para destruir região compartilhada.**

3) (1,5) Por que o conceito de processo é tão importante no projeto de sistemas multiprogramáveis? Qual é a estrutura de dados do sistema operacional que armazena as informações de um processo e como ela é dividida?

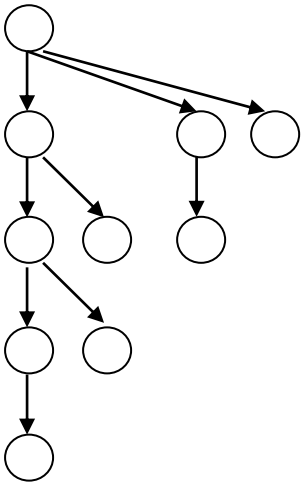
Resp.: Através de processos, um programa pode alocar recursos, compartilhar dados, trocar informações e sincronizar sua execução. Nos sistemas multiprogramáveis os processos são executados concorrentemente,

compartilhando o uso do processador, memória principal, dispositivos de E/S dentre outros recursos. A estrutura de dados do SO que armazena as informações de um processo é o PCB (Process Control Block) e é formada por três partes, conhecidas como contexto de hardware, contexto de software e espaço de endereçamento, que juntos mantêm todas as informações necessárias à execução de um programa.

4) (1,5) Considerando o trecho de código abaixo, responda:

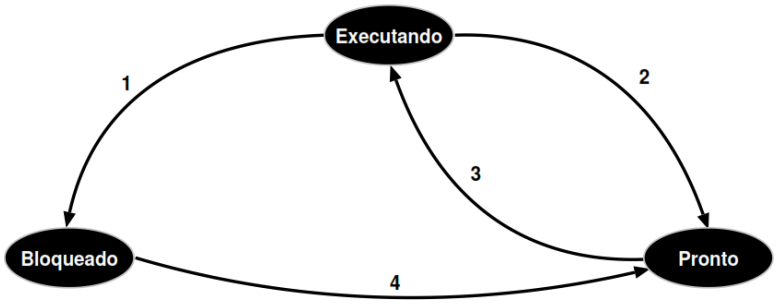
```
pid_t x;
x = fork();
if (x > 0) fork();
if (x == 0) x = fork();
fork();
if (x == 0) fork();
```

a) (1,0) Como fica a árvore de processos do processo inicial, com todos os filhos criados (considerando que nenhum deles terminou e assumindo que não houve falha na execução de nenhuma chamada fork())?



b) (0,5) Ao executar este trecho, quantos processos são criados, incluindo o processo inicial, assumindo que não houve falha na execução de nenhuma chamada fork()?
Resp.: 10

5) (3,0) Observe o diagrama de estados descrito na figura a seguir:

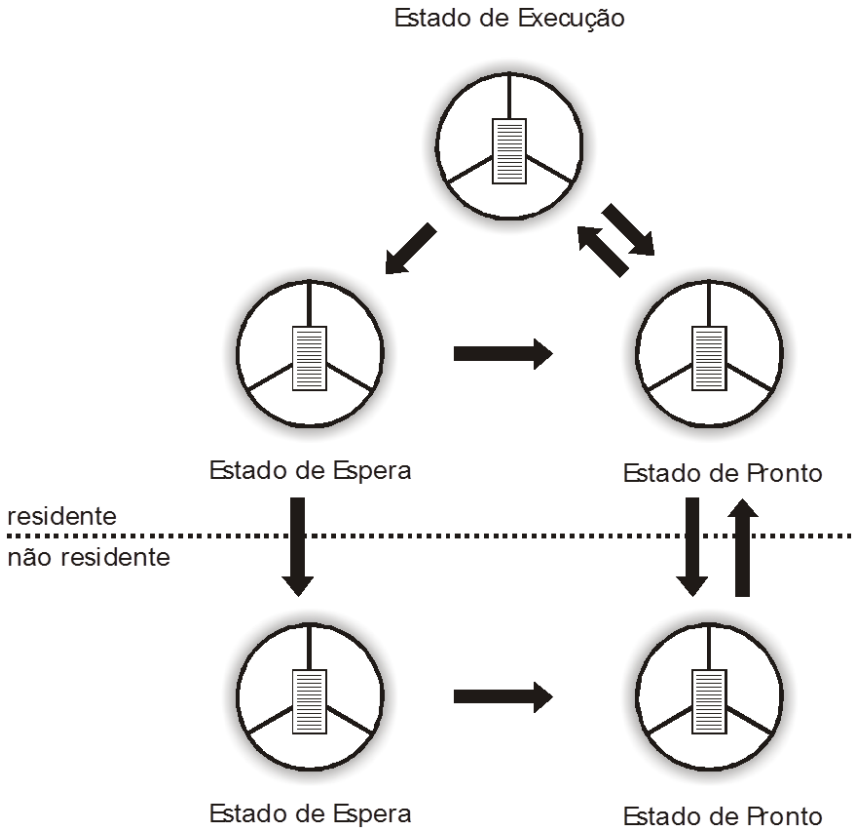


a) (1,5) Descreva os estados e o significado dos mesmos para o Sistema Operacional. Descreva também as transições explicando os possíveis motivos para que cada uma delas venha a ocorrer.

Resp.: Um processo pode estar em três estados: executando, quando este está sendo executado pelo processador; pronto, quando este processo está esperando para ser executado no processador (pois algum outro processo está em execução no processador); e bloqueado, quando o processo não pode executar no processador até a ocorrência de algum evento externo (como a finalização uma chamada de sistema). A Transição 1, do estado executando para o bloqueado, ocorre quando um processo em execução descobre que somente poderá continuar a executar após a ocorrência de um certo evento externo à sua execução (quando o processo faz uma chamada de sistema, por exemplo). A Transição 2, do estado executando para o pronto, ocorre quando o escalonador determinou que o processo atualmente em execução já executou por muito tempo no processador ou quando ocorre uma interrupção. A Transição 3, do estado pronto para o executando, ocorre quando o escalonador determinou que é a vez deste processo, que estava esperando pelo uso do processador, de

executar no processador por algum tempo. Finalmente, a Transição 4, do estado bloqueado para o pronto, significa que o evento externo pelo qual este processo estava esperando ocorreu, e com isso, o processo poderá agora ser escolhido pelo escalonador para ser futuramente executado pelo processador.

b) (1,5) Modifique o diagrama de estado para incluir os estados de Pronto no Disco e Bloqueado no Disco, explicando as novas transições e estados.



Resp.: Processos que estejam prontos ou em espera na memória (residentes) podem ser transferidos para o disco (não residentes), caso o SO precise de mais memória para outros processos. Este mecanismo também pode ser usado para reduzir o grau de multiprogramação do sistema (com um escalonador de médio prazo). Um processo em espera na memória pode passar para o estado "em espera no disco". Da mesma forma, um processo pronto na memória pode passar para o estado de "pronto no disco". Quando o evento esperado por um processo em espera no disco acontece, ele é passado para o estado de pronto no disco. Um processo pronto no disco pode ser selecionado pelo SO para ser movido novamente para a memória, passando então para o estado de pronto na memória, onde em um próximo momento poderá ser selecionado para execução.

6) (1,5) Descreva as diferenças entre os sistemas operacionais do tipo (i) monolítico; (ii) estruturado em camadas; e (iii) micronúcleo (cliente-servidor)

Resp.: Um sistema operacional monolítico, como o Linux, é caracterizado por não conter nenhuma estrutura interna no seu núcleo, isto é o núcleo é composto por um conjunto de procedimentos que gerenciam os dispositivos do hardware ou que executam as chamadas ao sistema. É um modelo mais caótico, pois não força nenhum tipo de organização na implementação. As partes diferentes do sistema podem se comunicar diretamente sem qualquer hierarquia. Já em um sistema operacional estruturado em camadas, como o UNIX, o núcleo do sistema é organizado em um conjunto de camadas, sendo que cada uma destas possui uma função específica, como a de gerenciar os dispositivos do hardware ou tratar das chamadas ao sistema. Neste modelo existe o overhead de se passar por pelas diferentes camadas para acessar o recurso desejado. A grande diferença entre um sistema operacional baseado em micronúcleo e os sistemas monolíticos ou baseados em camadas é a de que a maior parte do gerenciamento do sistema é feita no modo usuário, por processos servidores, que devem receber mensagens dos processos clientes para executar a sua função, como por exemplo, o tratamento de uma chamada ao sistema. O núcleo executa somente as funções mais básicas do sistema, como o tratamento da troca de mensagens entre clientes e servidores, e o acesso direto aos dispositivos físicos do hardware.