Q1a2

Q1.1 Comente a seguinte frase: "O sistema operativo é um computador virtual".

A frase "O sistema operativo é um computador virtual" refere-se à capacidade que o sistema operativo tem de esconder os detalhes complexos do hardware e apresentar aos utilizadores e programadores uma interface mais simples, uniforme e abstrata. No fundo, é como se o sistema operativo criasse uma versão virtual do computador real, mas mais amigável.

Por exemplo, em vez de obrigar os programas a aceder diretamente a discos, memórias, CPUs e controladores, o sistema operativo oferece conceitos abstratos como ficheiros, processos ou espaços de memória virtuais. Cada processo tem a ilusão de estar sozinho no sistema, com o seu próprio processador e memória, mesmo que esteja a partilhar esses recursos com outros.

O sistema operativo atua como uma máquina estendida, tornando o computador mais fácil de programar, e como um gestor de recursos, assegurando que os diferentes programas utilizam o hardware de forma justa e eficiente. Portanto, a metáfora utilizada do "computador virtual" é bastante adequada.

156 / 220 Word Limit

Q1.2 Um sistema operativo do tipo embutido encontra-se normalmente em que tipo de dispositivos?

Os sistemas operativos embutidos são tipicamente encontrados em dispositivos que não se destinam a ser computadores tradicionais, mas que mesmo assim precisam de controlar hardware e executar software. Estes sistemas são desenhados para serem pequenos, eficientes e altamente especializados para tarefas especificas.

Normalmente, encontramos sistemas operativos embutidos em dispositivos como electrodomesticos (fornos, máquinas de lavar, microondas), automóveis (controlo de travões, motores, sensores), equipamentos médicos (monitores cardíacos, bombas de insulina), sistemas industriais de controlo, câmaras de vigilância, routers, relógios inteligentes e até brinquedos eletrónicos.

Um ponto importante é que muitos destes dispositivos operam em tempo real ou com recursos muito limitados (memória, capacidade de processamento, energia), pelo que o sistema operativo embutido precisa de ser leve e fiável. Alguns exemplos de sistemas deste tipo são o VxWorks, Embedded Linux, FreeRTOS, entre outros.

Estes sistemas não costumam permitir que os utilizadores instalem programas arbitrários, o que os distingue dos sistemas mais generalistas, e estão geralmente otimizados para uma função muito específica, com grande foco na fiabilidade e na previsibilidade.

167 / 220 Word Limit

Q3a4

Q1.3 Relacione e caracterize os conceitos de multiprogramação e de partilha no tempo (timesharing).

A multiprogramação e a partilha no tempo são dois conceitos centrais no desenvolvimento dos sistemas operativos modernos, e embora estejam relacionados, tem objetivos diferentes. A multiprogramação surgiu primeiro e tem como principal finalidade maximizar a utilização do CPU. A ideia é manter vários processos na memória ao mesmo tempo, de forma que, quando um processo fica bloqueado (por exemplo, à espera de I/O), o sistema possa rapidamente passar para outro processo que esteja pronto a executar. Isto garante que o CPU não fique parado.

A partilha no tempo por sua vez, é uma evolução deste conceito, com foco na interatividade. Aqui, os processos dos utilizadores não só coexistem na memória, como também recebem pequenos períodos de tempo da CPU (chamados de quantums), alternando entre eles de forma tão rápida que dá a sensação de que estão todos a correr ao mesmo tempo. Este modelo é essencial para sistemas com utilizadores humanos, como terminais ou desktops, onde é importante que cada comando receba uma resposta quase imediata.

Concluindo, a multiprogramação procura eficiência e aproveitamento dos recursos, enquanto a partilha no tempo acrescenta a capacidade de resposta em ambientes interativos, partilhando ativamente o tempo do CPU entre vários utilizadores.

197 / 220 Word Limit

Q1.4 Indique justificando as três razões principais da existência de tarefas.

As tarefas (ou threads) existem principalmente para permitir maior eficiência, paralelismo e simplicidade no design de aplicações que executam múltiplas atividades em simultâneo. A primeira razão para a sua existência está na melhoria de desempenho. Criar múltiplas threads dentro de um mesmo processo é muito mais leve do que criar múltiplos processos, porque as threads partilham o mesmo espaço de memória e os mesmos recursos, evitando custos elevados de criação e troca de contexto.

A segunda razão prende-se com a estruturação do código. Ao dividir uma aplicação em várias threads, cada uma responsável por uma parte específica do trabalho (como a leitura de dados, o processamento e a escrita), o programa torna-se mais modular e mais fácil de entender, desenvolver e manter. Isto é especialmente útil em aplicações interativas, como servidores ou interfaces gráficas.

Por fim, a terceira razão é a capacidade de aproveitar melhor os sistemas multiprocessador. Com várias threads a correr em paralelo, é possivel distribuir a carga de trabalho por diferentes núclos de CPU, o que acelera significativamente a execução em ambientes com múltiplos processadores.

Concluindo, o modelo de threads é uma solução eficiente e flexível para a concorrência moderna.

193 / 220 Word Limit

Q5a6

Q1.5 Explique em que consiste o método da barreira (barrier) de sincronização entre processos.

O método da barreira é uma técnica de sincronização usada em ambientes concorrentes para garantir que um grupo de processos ou tarefas apenas avança para uma fase seguinte quando todos os elementos do grupo tiverem atingido um certo ponto de execução. É como se todos tivessem de "esperar uns pelos outros" antes de continuar.

Por exemplo, um programa que divide o trabalho em várias partes e atribui cada parte a uma thread. Após cada thread terminar a sua parte, todas têm de aguardar até que as outras também tenham concluido. Só depois é que podem avançar em conjunto para a próxima etapa. Este ponto de espera chama-se barreira.

A barreira é útil, por exemplo, em algoritmos paralelos que se organizam em ciclos ou fases, e onde é essencial manter as threads sinccronizadas. A implementação pode ser feita com variáveis de condição, semáforos ou outras primitivas, e requer uma contagem de quantas tarefas chegaram à barreira, só libertando todas quando esse número for atingido.

163 / 220 Word Limit

Q1.6 Observe o seguinte programa com dois processos:

```
/* pseudo-codigo */
semaforo recurso1, recurso2;

void processoA(void){
  down(&recurso1);
  down(&recurso2);
  utilizar_ambos_os_recursos();
  up(&recurso2);
  up(&recurso1);
}

void processoB(void){
  down(&recurso2);
  down(&recurso1);
  utilizar_ambos_os_recursos();
  up(&recurso1);
  up(&recurso2);
}
```

Este programa pode potencialmente levar a uma situação de impasse (deadlock)? Justifique.

Este programa pode levar a uma situação de impasse sim, porque os dois processos tentam obter acesso a dois recursos partilhados, mas em ordens diferentes. O processo A primeiro faz down no recurso1 e depois no recurso2, enquanto o processo B faz down no recurso2 e só depois no recurso1.

Isto cria uma situação clássica em que o processo A pode ficar à espera do recurso2 que já foi ocupado pelo processo B, enquanto o processo B fica à espera do recurso1, já ocupado pelo processo A. Nenhum deles consegue continuar porque ambos estão à espera um do outro e esta dependência circular caracteriza um deadlock.

Para que ocorra um deadlock é necessário uma das quatro condições: exclusão mútua, posse e espera, não-preempção e espera circular. No código apresentado, todas estas condições estão presentes, o que torna a situação de impasse possível. Para evitar isto, seria necessário garantir uma ordem fixa de aquisição de recursos, ou aplicar uma estratégia como o protocolo do banqueiro ou a prevenção da espera circular.

170 / 220 Word Limit

Q7a8

Q1.7 No âmbito da memória virtual, a que corresponde o conceito de moldura de página (page frame) ou página física?

No contexto da memória virtual, uma modulra de página (ou page frame) corresponde a uma unidade de memória física na RAM, com tamanho fixo, que pode ser utilizada para armazenar uma página de memória virtual. O conceito de página física surge como par da página virtual, enquanto a virtual é usada pelos processos e pelo sistema operativo para abstrair o acesso à memória, a física é o espaço real onde os dados residem.

A RAM do sistema é dividida em múltiplas molduras, todas com o mesmo tamanho, e quando uma página virtual é carregada para a memória, ela é colocada numa destas molduras. O sistema operativo, com a ajuda da unidade de gestão de memória (MMU), mantém uma tabela de páginas onde associa cada página virtual ao número da moldura física correspondente.

Este mecanismo é essencial para a implementação da memória virtual, pois permite que cada processo tenha a ilusão de um espaço de memória contínuo e privado, mesmo que, na realidade, as suas páginas estejam dispersas na RAM ou até mesmo em disco.

174 / 220 Word Limit

Q1.8 Considere um sistema de memória virtual com um esquema de Tabela de Páginas mononível. Supondo que não ocorre uma falha de página, descreva qualitativamente como é obtido o endereço físico a partir do endereço virtual. Considere páginas de igual dimensão.

Num sistema de memória virtual com uma Tabela de Páginas mononível, a tradução de um endereço virtual para um endereço físico faz-se em dois passos principais: indentificar a página virtual e depois localizar a moldura física onde essa página se encontra.

O endereço virtual é normalmente dividido em duas partes: os bits mais à esquerda representam o número da página virtual (virtual page number), e os bits mais à direita representam o offset dentro dessa página. Como todas as páginas têm o mesmo tamanho, o offset tem sempre o mesmo número de bits e indica exatamente onde, dentro da página, está o dado pretendido.

O sistema operativo (com a MMU) consulta então a Tabela de Páginas, usando o número da página virtual como índice. Esta tabela diz em que moldura física da RAM essa página está armazenada. Finalmente, o endereço físico é construído ao combinar o número da moldura (frame number) com o mesmo offset original, ou seja, apontando para a mesma posição dentro da nova página, agora em memória real.

Este processo é rápido e transparente para o processo que está a correr, e é um dos pilares fundamentais da separação entre espaço de endereçamento virtual e físico.

199 / 220 Word Limit

Q9a10

Q1.9 Para um sistema de ficheiros baseado em i-nodes, explique no que consiste e como é implementado um "hard link".

Num sistema de ficheiros baseado em i-nodes (como o usado em muitos sistemas UNIX), cada ficheiro é representado por uma estrutura chamada i-node, que contém todas as informações sobre o ficheiro: permissões, autor, datas e os apontadores para os blocos de dados no disco. Os nomes dos ficheiros, por outro lado, existem em diretórios e funcionam como referências (ou links) para os i-nodes.

Um hard link é uma forma de criar uma segunda entrada (ou nome) que aponta para o mesmo i-node de um ficheiro já existente. Ou seja, não se cria uma cópia dos dados, cria-se apenas outro caminho para aceder exatamente ao mesmo conteúdo. Ambas as entradas partilham o mesmo contador de ligações (link count) e, enquanto esse número for maior do que zero, o i-node e os dados associados continuam válidos.

Esta técnica é útil, por exemplo, para manter acesso a um ficheiro mesmo que o nome original seja apagado. O ficheiro só é realmente eliminado quando todos os hard links forem removidos.

166 / 220 Word Limit

Q1.10 Explique o conceito de I/O por mapeamento de memória.

O conceito de I/O por mapeamento de memória consiste em tratar os dispositivos de entrada/saída como se fossem áreas da memória. Em vez de usar instruções especiais de I/O, o sistema mapeia os registos de controlo dos dispositivos diretamente para endereços dentro do espaço de endereçamento. Assim, ler ou escrever num dispositivo torna-se tão simples como aceder a uma variável na memória.

Isto funciona porque o sistema operativo (com a ajuda da MMU) reserva zonas de memória física que estão associadas a dispositivos, como por exemplo, um controlador de disco ou uma porta série. Quando o processador acede a esses endereços, não está realmente a comunicar com a RAM, mas sim com o hardware do dispositivo.

Esta abordagem simplifica a comunicação com dispositivos e permite usar o mesmo conjunto de instruções da CPU para memória e periféricos. Também pode oferecer ganhos de desempenho, especialmente quando se quer evitar cópias desnecessárias de dados entre buffers.

154 / 220 Word Limit