Exercício

1) Como uma aplicação pode implementar concorrência em um ambiente monothread?

Através de processos independentes e subprocessos

2) Quais os problemas de aplicações concorrentes desenvolvidas em ambientes monothread?

Um problema é que o uso de processos no desenvolvimento de aplicações concorrentes demanda consumo de diversos recursos do sistema. Sempre que um novo processo é criado, o sistema deve alocar recursos para cada processo, consumindo tempo de processador neste trabalho. No caso do término do processo, o sistema dispensa tempo para desalocar recursos previamente alocados.
Outro problema a ser considerado é quanto ao compartilhamento do espaço de endereçamento. Como cada processo possui seu próprio espaço de endereçamento, a comunicação entre processos torna-se difícil e lenta, pois utiliza mecanismos como pipes, sinais, semáforos, memória compartilhada ou troca de mensagem.

3) O que é um ambiente multithread e quais as vantagens de sua utilização?

Um thread pode ser definido como uma subrotina de um programa que pode ser executada de forma assíncrona, ou seja, executada paralelamente ao programa chamador. A grande vantagem no uso de threads é a possibilidade de minimizar a alocação de recursos do sistema, além de diminuir o overhead na criação, troca e eliminação de processos.

4) Explique a diferença entre unidade de alocação de recursos e unidade de escalonamento.

Em ambientes monothread, o processo é ao mesmo tempo a unidade de alocação de recursos e a unidade de escalonamento. A independência entre os conceitos de processo e thread permite separar a unidade de alocação de recursos da unidade de escalonamento, que em ambientes monothread estão fortemente relacionadas. Em um ambiente multithread, a unidade de alocação de recursos é o processo, onde todos os seus threads compartilham o espaço de endereçamento, descritores de arquivos e dispositivos de E/S. Por outro lado, cada thread representa uma unidade de escalonamento independente e, neste caso, o sistema não seleciona um processo para a execução, mas sim um de seus threads.

5) Quais as vantagens e desvantagens do compartilhamento do espaço de endereçamento entre threads
de um mesmo processo?

Como threads de um mesmo processo compartilham o mesmo espaço de endereçamento, não existe qualquer proteção no acesso à memória, permitindo que um thread possa alterar facilmente dados de outros. Para que threads trabalhem de forma cooperativa, é fundamental que a aplicação implemente mecanismos de comunicação e sincronização entre threads, a fim de garantir o acesso seguro aos dados compartilhados na memória. Por outro lado, o compartilhamento do espaço de endereçámento é extremamente simples e rápido.

6) Compare os pacotes de threads em modo usuário e modo kernel.

Threads em modo usuário (TMU) são implementados pela aplicação e não pelo sistema operacional. Para isso, deve existir uma biblioteca de rotinas que possibilita à aplicação realizar tarefas como criação/eliminação de threads, troca de mensagens entre threads e uma política de escalonamento. Neste modo, o sistema operacional não sabe da existência de múltiplos threads, sendo responsabilidade exclusiva da aplicação gerenciar e sincronizar os diversos threads existentes.
Threads em modo kernel (TMK) são implementadas diretamente pelo núcleo do sistema operacional, através de chamadas a rotinas do sistema que oferecem todas as funções de gerenciamento e sincronização. O sistema operacional sabe da existência de cada thread e pode escaloná-los individualmente. No caso de múltiplos processadores, os threads de um mesmo processo podem ser executados simultaneamente.

7) Qual a vantagem do scheduler activations comparado ao pacote híbrido?

A principal vantagem é melhorar o desempenho no seu uso evitando as mudanças de modos de acesso desnecessárias (usuário-kernel-usuário).
Caso um thread utilize uma chamada ao sistema que o coloque no estado de espera, não é necessário que o kernel seja ativado, bastando que a própria biblioteca em modo usuário escalone outro thread. Isto é possível porque a biblioteca em modo usuário e o kernel se comunicam e trabalham de forma cooperativa. Cada camada implementa seu escalonamento de forma independente, porém trocando informações quando necessário.

8) Dê exemplos do uso de threads no desenvolvimento de aplicativos, como editores de textos e planilhas eletrônicas.

9)

9) Como o uso de threads pode melhorar o desempenho de aplicações paralelas em ambientes com múltiplos processadores?

Para obter os benefícios do uso de threads, uma aplicação deve permitir que partes diferentes do seu código sejam executadas em paralelo de forma independente. O uso de uma arquitetura com múltiplos processasdores beneficia a concorrência entre os threads com a possibilidade do paralelismo de execução entre processadores.

10) Quais os benefícios do uso de threads em ambientes cliente-servidor?

O principal benefício do uso de threads em ambientes cliente-servidor é a melhoria no desempenho da aplicação servidora. Além disso, a comunicação entre os threads no servidor pode ser feita através de mecanismos mais simples e eficientes.

11) Como o uso de threads pode ser útil em arquiteturas microkernel?

A arquitetura microkernel utiliza processos para implementar funções relativas ao kernel do sistema operacional, sendo que esses processos são utilizados como servidores quando algum cliente necessita de algum serviço do sistema. Arquiteturas que implementam threads, possibilitam um melhor desempenho dos processos servidores.