Cap 03
Processos

1. O que são Processos? Como são classificados? Qual a diferença entre um Processo e um Programa? Explique;

Um processo é a representação em execução de um programa, com seu estado atual e recursos associados, permitindo a execução simultânea de várias tarefas. Processos podem ser classificados pelo estado (pronto, executando, bloqueado), relação (pai/filho, independente), prioridade (alta/baixa) e tempo de vida (temporário, persistente). Essa diferenciação é crucial para entender a distinção entre processos e programas: um programa é o código armazenado, enquanto um processo é a execução desse código, incorporando sua dinâmica e contexto.

2. Quais são os possíveis estados de um Processo? Explique sucintamente como o processo se encontra em cada estado;

Os estados de um processo são: 1) Pronto - aguarda execução, com recursos carregados; 2) Executando - ativamente processando instruções; 3) Bloqueado - espera por eventos externos, liberando a CPU. A transição entre esses estados depende da interação do processo com o sistema operacional e eventos, permitindo um gerenciamento eficiente dos recursos e multitarefa.

3. O que são ***Filas de Scheduling***? Como são subdivididas? Explique;

Filas de escalonamento são estruturas usadas em sistemas operacionais para gerenciar a execução de processos em um ambiente multitarefa. Elas são subdivididas em várias categorias, incluindo a fila de pronto para processos prontos para execução, a fila de bloqueados para processos temporariamente impedidos por eventos externos, a fila de suspensos para processos removidos da memória, a fila de terminados para processos finalizados e a fila de prioridades para processos classificados por níveis de prioridade. Essas subdivisões permitem um gerenciamento eficiente e organizado dos processos, otimizando o uso dos recursos da CPU e a execução justa dos processos concorrentes

4. O que você entende por ***Context Switching***? E ***Overhead***?

Context Switching: Troca de contexto envolve salvar o estado atual de um processo em execução (registradores da CPU, ponteiros de pilha, etc.) e carregar o estado de um novo processo, permitindo multitarefa. Isso possibilita uma execução aparentemente simultânea de processos, mas introduz algum overhead devido ao tempo e recursos necessários para a troca.

Overhead: Overhead é o custo adicional ou recursos consumidos por atividades que não contribuem diretamente para a tarefa principal. Em sistemas operacionais, o overhead ocorre durante operações como a troca de contexto, essencial para a multitarefa, mas que consome tempo da CPU e recursos de memória, afetando o desempenho do sistema. Minimizar o overhead é crucial para uma operação eficiente.

5. Como o Sistema Operacional consegue gerenciar cada processo em execução de forma adequada? Que tipo de estrutura é mantida por ele a fim de prover esta organização?

O sistema operacional gerencia cada processo em execução por meio de uma Tabela de Processos (PCB) que armazena detalhes como estado, registradores e informações de recursos. Além disso, utiliza filas de escalonamento para determinar a ordem de execução, empregando algoritmos como Round Robin e First-Come, First-Served. O sistema operacional também regula o acesso a recursos compartilhados, usando técnicas de sincronização e semáforos. Essa infraestrutura complexa de informações, estruturas de dados e algoritmos permite o gerenciamento adequado, justo e eficiente dos processos, facilitando a multitarefa e a operação tranquila do sistema.

6. Como é estruturado um Processo na memória Principal? Explique em detalhes;

Um processo na memória principal é estruturado em várias seções: o código contém instruções executáveis, os dados inicializados armazenam variáveis globais e estáticas, a seção BSS guarda variáveis não inicializadas, o heap permite alocação dinâmica de memória, e a pilha mantém informações de contexto e variáveis locais de funções. Essa organização permite ao sistema operacional gerenciar eficientemente a execução do processo, garantindo a integridade e a eficiência no uso da memória

7. O que são **processos filhos**? Por quais razões um processo pai pode finalizar um processo filho?

Processos filhos são criados por processos pais para realizar tarefas específicas. Um processo pai pode finalizar um processo filho por razões como conclusão de tarefas, erros na execução, economia de recursos ou conformidade com políticas de gerenciamento do sistema. Isso permite ao processo pai controlar eficientemente os recursos e as atividades em execução no sistema.

8. O que você entende por ***Sockets***? Quando são utilizados?

Sockets são interfaces de programação usadas para permitir a comunicação entre processos em diferentes computadores por meio de redes, como a Internet. Eles são empregados para criar aplicações cliente-servidor, transferir dados em tempo real, possibilitar comunicações em tempo real, oferecer acesso remoto e suportar aplicações distribuídas. Os sockets viabilizam a troca eficiente e confiável de dados entre processos através de conexões de rede, permitindo a construção de diversas aplicações de rede.

9. Diferentes processos podem se comunicar entre si? Em caso afirmativo, quais são as técnicas utilizadas para prover esta comunicação e em que situações devem ser utilizadas?

Sim, diferentes processos podem se comunicar em um sistema operacional, utilizando diversas técnicas adequadas a diferentes situações. Essas técnicas incluem o compartilhamento de memória, sockets para comunicação através de redes, pipes e FIFOs para comunicação entre processos locais, sinais para notificações assíncronas, memória compartilhada com semáforos para acesso coordenado, RPC para chamadas de procedimento entre processos remotos e message passing para troca flexível de mensagens. A escolha da técnica depende dos requisitos específicos da aplicação, como eficiência, sincronização, isolamento e comunicação interprocessual ou intercomputacional.