**Questionário 1**  
  
**Considerando as diversas definições de Sistema Operacional, opine se um sistema operacional deveria – ou não – incluir aplicações como navegadores Web ou programas de e-mail. Justifique sua resposta.**

Depende do uso. Em empresas por exemplo, um sistema operacional como o Windows que já possui vários programas de uso geral, acaba sendo mais vantajoso. Por outro lado um computador para uso pessoal com um sistema operacional sem aplicativos por padrão acaba sendo uma escolha melhor para poder personalizar ao seu gosto.

**Quais as principais funções de um sistema operacional? Cite exemplos.**

As principais funções de um sistema operacional é fazer o gerenciamento de várias partes de um computador como recursos, memórias, processos e dispositivos. Por exemplo, o fornecimento de drivers e a coordenação das operações dos dispositivos, para mouses, impressoras, teclados, etc.

**Considerando que o computador trabalha em linguagem de máquina ou binária (de baixo nível), comente por que a maioria dos programadores não programam em linguagem de baixo nível.**

Um dos motivos de programadores não programarem em alto nível é pelo fato de que quando o código é escrito em linguagem de alto nível é mais fácil de manter, revisar e depurar. Além de serem linguagens que tornam a programação mais rápida devido ao uso de frameworks e bibliotecas.

**Visto que os seres humanos trabalham melhor com linguagens de alto nível, comente por que os computadores continuam sendo fabricados em linguagens de baixo nível.**

Os computadores continuam sendo fabricados em linguagem de baixo nível, pois eles utilizam o sistema de camadas onde uma linguagem de alto nível é traduzida para uma linguagem de máquina pelo sistema operacional, já que é mais pratico produzir computadores que entendem apenas 1 e 0 (com ou sem energia).

**Defina o conceito de uma máquina de níveis ou camadas.**

Uma máquina de níveis ou camadas são maquinas que possuem softwares como sistemas operacionais entre outros que divide um sistema complexo em componentes ou camadas bem definidas, cada uma com um conjunto específico de responsabilidades.

**Questionário 2**

**Qual a grande diferença entre sistemas monoprogramáveis e sistemas multiprogramáveis?**

Nos sistemas monoprogramáveis, também conhecidos como sistemas de lote simples, apenas um único programa é executado por vez, isso significa que a CPU e outros recursos do sistema são dedicados exclusivamente à execução desse programa até que ele seja concluído, após a conclusão, o próximo programa na fila é carregado e executado, essa abordagem é simples, mas pode resultar em desperdício de recursos, uma vez que a CPU e outros componentes podem ficar ociosos durante a espera pela conclusão do programa em execução.

Por outro lado, nos sistemas multiprogramáveis, vários programas (ou processos) podem ser carregados na memória ao mesmo tempo e alternados rapidamente pela CPU, isso permite que múltiplos programas compartilhem os recursos do sistema, como CPU, memória e dispositivos de entrada/saída, a técnica de comutação rápida entre processos cria a ilusão de que vários programas estão sendo executados simultaneamente, embora a CPU esteja alternando entre eles em intervalos muito pequenos.

**Quais as vantagens dos sistemas multiprogramáveis?**

Os sistemas multiprogramáveis oferecem vantagens notáveis em comparação com sistemas monoprogramáveis, eles permitem a execução simultânea de vários programas, otimizando o uso da CPU e dos recursos, isso resulta em maior eficiência, multitarefa ágil e melhor experiência do usuário, especialmente em ambientes de compartilhamento de tempo, a produtividade é aumentada, sendo ideais para processamento em lote. Além disso, esses sistemas são mais resilientes a falhas, garantindo a continuidade de processos mesmo em caso de problemas, em resumo, os sistemas multiprogramáveis proporcionam eficiência, agilidade e melhor utilização de recursos, beneficiando a computação em diversos cenários.

**Um sistema monousuário pode ser um sistema multiprogramável? Dê um exemplo.**

Sim, um sistema monousuário pode ser transformado em um sistema multiprogramável, permitindo a execução simultânea de vários aplicativos ou tarefas por um único usuário, a transição envolve habilitar a alternância entre processos, melhorando a eficiência e a utilização dos recursos do sistema, portanto, a distinção entre monousuário e multiprogramável não está ligada apenas ao número de usuários, mas à capacidade de executar tarefas simultaneamente. Um exemplo é um computador pessoal em casa, onde um único usuário utiliza o sistema para suas tarefas diárias, o usuário inicia sessão, executa um processador de texto para escrever um documento, fecha o processador de texto e, em seguida, abre um navegador para acessar a internet.

**Quais são os tipos de sistemas multiprogramáveis?**

Existem vários tipos de sistemas multiprogramáveis, que são projetados para permitir a execução simultânea de múltiplos processos e o compartilhamento eficiente dos recursos do sistema, os principais tipos incluem sistemas Batch, onde tarefas são executadas em lotes sem necessidade de interação constante; sistemas de Tempo Compartilhado (Time Sharing), que possibilitam que múltiplos usuários interajam com o sistema em fatias de tempo, promovendo a multitarefa; sistemas de Tempo Real, que são projetados para responder a eventos dentro de prazos específicos, sendo essenciais para aplicações de controle em tempo real; sistemas Distribuídos, nos quais várias máquinas se comunicam e colaboram por meio de redes, proporcionando escalabilidade e resiliência; sistemas de Processamento Paralelo, que executam tarefas simultaneamente em múltiplas CPUs ou núcleos, visando a melhoria do desempenho para cargas de trabalho paralelizáveis; e sistemas Embarcados, otimizados para dispositivos específicos como dispositivos médicos, smartphones e outros dispositivos eletrônicos, com ênfase na eficiência de recursos e requisitos específicos do dispositivo. A escolha do tipo de sistema depende das demandas e características da aplicação, bem como das capacidades do hardware disponível, pois cada tipo possui suas próprias vantagens e desafios únicos.

**Pesquise na Internet algum Sistema Operacional monotarefa.**

Um exemplo de sistema operacional monotarefa é o MS-DOS (Microsoft Disk Operating System), o MS-DOS foi um sistema operacional popular na década de 1980 e início da década de 1990, amplamente utilizado em computadores pessoais da IBM e compatíveis, ele era projetado principalmente para executar uma tarefa por vez e não possuía recursos avançados de multitarefa, como os sistemas operacionais modernos. As aplicações eram executadas uma de cada vez, e os usuários precisavam fechar um programa antes de abrir outro.

**Pesquise na Internet dez Sistemas Operacionais multitarefas.**

Windows, Linux, macOS, Android, iOS, Windows Server, UNIX, Chrome OS, AIX, QNX

**Dos sistemas pesquisados na atividade anterior, informe quais são multiusuários e quais são monousuários.**

Multiusuários: Windows (em ambientes de servidor), Linux (Ubuntu, Fedora e outras distribuições), macOS, Android, iOS, Windows Server, UNIX (FreeBSD, Solaris e outras variantes), Chrome OS, AIX, QNX.

Monousuários: Windows

**Questionário 3**

**Pesquise o que é concorrência e diga como esse conceito está presente nos sistemas operacionais multiprogramáveis?**

A concorrência é a execução simultânea de várias tarefas, como processos ou threads, em um sistema de computador. Nos sistemas operacionais multiprogramáveis, a concorrência é fundamental para melhorar o desempenho e a utilização de recursos. Ela é alcançada através do escalonamento de processos, suporte a multithreading e gerenciamento eficiente de recursos compartilhados. A concorrência também envolve a comunicação entre entidades concorrentes para coordenação e troca de informações.

**Explique o mecanismo de funcionamento das interrupções.**

Interrupções são um mecanismo em sistemas de computadores que permitem que a CPU seja notificada de eventos específicos. Quando um evento ocorre, a CPU salva seu estado atual, identifica a fonte da interrupção, executa um tratador de interrupção específico para lidar com o evento e, em seguida, restaura seu estado anterior para continuar a execução do programa interrompido. Isso melhora a capacidade de resposta e eficiência dos sistemas.

**Qual a vantagem da E/S controlada por interrupção, comparada com a técnica de polling?**

A E/S controlada por interrupção é mais eficiente e econômica em termos de uso de recursos da CPU do que a técnica de polling. Ela permite que a CPU continue a execução de outras tarefas, respondendo apenas quando uma interrupção é gerada em eventos de E/S, tornando o sistema mais responsivo e eficiente. Por outro lado, o polling é mais simples e oferece controle direto sobre quando verificar o status de E/S, sendo útil em situações específicas, mas geralmente menos eficiente. A E/S controlada por interrupção é a abordagem preferida em sistemas modernos, enquanto o polling é utilizado em cenários específicos.

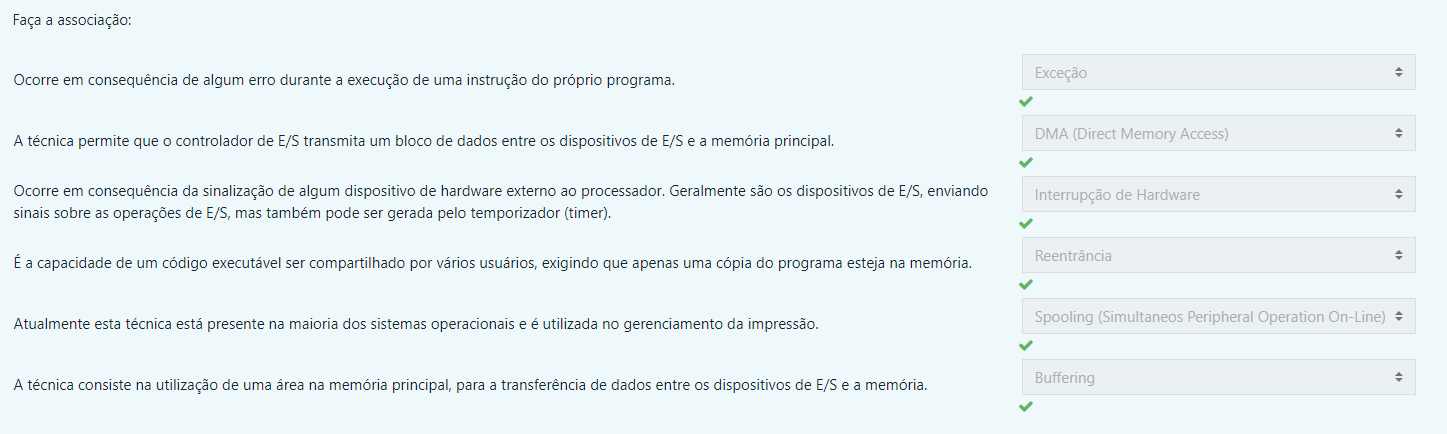
**O que é DMA e qual a vantagem desta técnica?**

O DMA (Acesso Direto à Memória) é uma técnica que permite que dispositivos de E/S acessem diretamente a memória principal, sem a intervenção constante da CPU. Suas vantagens incluem melhoria de desempenho, economia de recursos de CPU, menor latência, suporte a grandes volumes de dados, melhoria da eficiência do sistema e potencial redução do consumo de energia. O DMA é particularmente útil em cenários com alto volume de transferência de dados e em sistemas que buscam otimizar o desempenho e a utilização de recursos.

**Em um sistema multiprogramável, seus usuários utilizam o mesmo editor de textos (200 Kb), compilador (300 Kb), software de correio eletrônico (200 Kb) e uma aplicação corporativa (500 Kb). Caso o sistema não implemente reentrância, qual o espaço de memória principal ocupado pelos programas, quando 10 usuários estiverem utilizando todas as aplicações simultaneamente? Qual o espaço liberado, quando o sistema implementa reentrância em todas as aplicações?**

Sem reentrância, 10 usuários utilizando todas as aplicações simultaneamente ocuparão 12 MB de memória principal, uma vez que cada instância de programa tem sua própria área de memória separada.

Com reentrância implementada, os programas compartilham o mesmo espaço de memória para código e dados não específicos de cada instância, economizando espaço, embora dados específicos do usuário ainda sejam alocados separadamente. Isso leva a uma utilização mais eficiente da memória e melhora a eficiência do sistema. A economia exata depende da quantidade de programas em execução e do número de usuários.



**Questionário 4**

**O que é o núcleo do sistema e quais são suas principais funções?**

O núcleo do sistema, também conhecido como kernel, é a parte central de um sistema operacional. Suas principais funções incluem gerenciar recursos de hardware, como CPU e memória, fornecer uma interface para aplicativos e serviços, e controlar o acesso a recursos compartilhados. Também é responsável pelo agendamento de tarefas, gerenciamento de processos e comunicação entre componentes do sistema.

**O que é uma system call e qual sua importância para a segurança do sistema? Como as system calls são utilizadas por um programa?**

Uma system call é uma interface que permite que os programas de usuário solicitem serviços específicos do sistema operacional. Esses serviços podem incluir operações de entrada e saída, acesso a hardware, criação e gerenciamento de processos, entre outros. As system calls são uma forma segura e controlada de os programas de usuário interagirem com o núcleo do sistema.

A importância das system calls para a segurança do sistema reside no fato de que elas atuam como uma camada de proteção entre os programas de usuário e o núcleo do sistema. As system calls permitem que o sistema operacional verifique e controle as solicitações dos programas de usuário, garantindo que eles não realizem operações não autorizadas ou prejudiciais ao sistema.

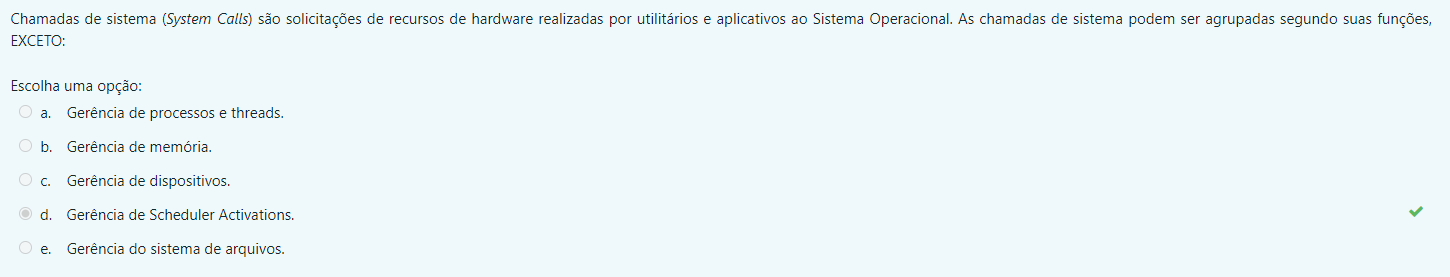
Os programas utilizam as system calls por meio de bibliotecas específicas fornecidas pelo sistema operacional ou por linguagens de programação que oferecem suporte a essas chamadas. Os programadores escrevem código de usuário que chama as funções apropriadas da biblioteca ou da linguagem, e essas funções, por sua vez, interagem com as system calls correspondentes do sistema operacional para realizar as operações desejadas. Essa abstração protege os programas de usuário de detalhes complexos e potencialmente perigosos do funcionamento interno do sistema operacional.

**O que são instruções privilegiadas e não privilegiadas? Qual a relação dessas instruções com os modos de acesso?**

Instruções privilegiadas são instruções de máquina que só podem ser executadas em um modo privilegiado ou modo supervisor, geralmente reservado para o kernel ou sistema operacional. Essas instruções têm acesso irrestrito aos recursos do sistema e podem executar operações críticas, como controlar hardware e modificar tabelas de páginas de memória.

Instruções não privilegiadas são instruções que podem ser executadas em um modo não privilegiado ou modo de usuário, que é usado para a execução de aplicativos normais. Essas instruções têm acesso limitado aos recursos do sistema e não podem executar operações sensíveis ao sistema, como manipulação direta de hardware.

A relação com os modos de acesso está relacionada ao controle de acesso aos recursos do sistema. Os sistemas operacionais geralmente implementam dois modos de execução: modo privilegiado (kernel mode) e modo não privilegiado (user mode). O sistema operacional executa em modo privilegiado, onde pode executar instruções privilegiadas e acessar recursos de hardware. Os aplicativos de usuário são executados em modo não privilegiado, onde têm acesso apenas às instruções não privilegiadas e a um conjunto limitado de recursos. Isso ajuda a proteger a integridade e a segurança do sistema, pois impede que aplicativos normais realizem operações potencialmente perigosas ou causem danos ao sistema.



**Compare as arquiteturas monolíticas e de camadas. Quais as vantagens e desvantagens de cada arquitetura?**

A arquitetura monolítica é simples e eficiente, mas limita a escalabilidade e a manutenção em projetos maiores. Por outro lado, a arquitetura em camadas oferece modularidade, escalabilidade e controle de recursos, mas pode ser mais complexa e demorada de desenvolver. A escolha depende das necessidades do projeto: monolítica para simplicidade, em camadas para complexidade e escalabilidade.

**Quais as vantagens do modelo de máquina virtual?**

O modelo de máquina virtual oferece vantagens como isolamento, portabilidade, consolidamento de recursos, facilidade em testes e recuperação de desastres, isolamento de falhas, flexibilidade em sistemas operacionais, escalabilidade e segurança. Isso simplifica a manutenção e o gerenciamento de sistemas, tornando as máquinas virtuais valiosas em diversas situações de computação.

**Como funciona o modelo cliente-servidor na arquitetura microkernel? Quais as vantagens e desvantagens dessa arquitetura?**

Na arquitetura microkernel, o modelo cliente-servidor divide o sistema operacional em servidores independentes para componentes como drivers e sistemas de arquivos, enquanto o núcleo fornece serviços essenciais. Vantagens incluem modularidade e isolamento de falhas, mas há desvantagens como overhead de comunicação e complexidade. A escolha depende dos requisitos do sistema e trade-offs entre flexibilidade e desempenho.

**Questionário 5**

**Como uma aplicação pode implementar concorrência em um ambiente monothread?**

Para implementar concorrência em um ambiente monothread, pode-se adotar estratégias como programação assíncrona, filas de tarefas, loop de eventos e simulação de threads. Dividir tarefas longas, usar temporizadores e adotar modelos orientados a eventos também são práticas úteis. Embora essas abordagens permitam a execução simultânea de tarefas, é fundamental gerenciar cuidadosamente possíveis problemas de concorrência, como condições de corrida, para garantir um funcionamento eficiente e sem erros.

**Quais os problemas de aplicações concorrentes desenvolvidas em ambientes monothread?**

Aplicações concorrentes em ambientes monothread enfrentam desafios significativos, incluindo bloqueio de thread única, baixa escalabilidade, limitações na responsividade, complexidade de programação, coordenação dificultada entre tarefas e dificuldades na depuração. A falta de threads separadas também impede a eficiente utilização de recursos do sistema. Ao desenvolver nesses ambientes, é crucial escolher estratégias adequadas para mitigar esses problemas específicos.

**O que é um thread e quais as vantagens em sua utilização?**

Threads são unidades básicas de execução em programas, permitindo concorrência e melhorando a eficiência. Vantagens incluem responsividade, compartilhamento de recursos, economia de recursos, rápida comutação de contexto, escalabilidade e programação modular. Elas são cruciais para operações I/O-bound e tarefas paralelas. No entanto, requerem cuidado com problemas de concorrência.

**Explique a diferença entre unidade de alocação de recursos e unidade de escalonamento?**

A unidade de alocação de recursos refere-se à entidade que solicita e consome recursos do sistema, como processos ou threads. Em contraste, a unidade de escalonamento é a menor entidade considerada pelo sistema operacional ao decidir a ordem de execução e alocação de recursos, podendo ser uma tarefa, processo ou thread. Enquanto a alocação de recursos é realizada com base em políticas específicas, o escalonamento envolve a decisão de como eficientemente alocar os recursos disponíveis. Ambas são essenciais para a gestão eficaz de recursos em um sistema operacional.

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

O compartilhamento do espaço de endereçamento entre threads do mesmo processo oferece eficiência na comunicação e coordenação, economizando recursos e simplificando o desenvolvimento. No entanto, isso introduz desafios como problemas de concorrência e sincronização, podendo resultar em corrupção de dados. A escolha entre essa abordagem e outras depende das necessidades específicas da aplicação.

**Questionário 6**

**Compare os pacotes de threads em modo usuário e em modo kernel?**

Threads em modo usuário são gerenciadas pela aplicação, mais leves em termos de desempenho, mas têm acesso limitado a recursos do sistema. Threads em modo kernel são gerenciadas pelo sistema operacional, mais eficientes em recursos, porém com maior custo de troca de contexto. A escolha depende das necessidades específicas do aplicativo, considerando desempenho, segurança, escalabilidade e portabilidade.

**Qual a vantagem do scheduler activations comparado ao pacote híbrido?**

O "scheduler activations" oferece maior eficiência na troca de contexto, escalabilidade e controle fino em comparação com um pacote de threads híbrido, tornando-o uma escolha vantajosa em ambientes de alta concorrência. No entanto, a portabilidade pode ser um desafio, já que depende de características específicas do sistema operacional. A escolha entre as duas abordagens depende das necessidades do aplicativo e das considerações de portabilidade.

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

Em editores de texto e planilhas eletrônicas, as threads são usadas para melhorar o desempenho e a responsividade. Em editores de texto, elas podem ser empregadas para carregar documentos, análise de sintaxe e verificação ortográfica em segundo plano. Em planilhas, as threads aceleram cálculos de fórmulas, atualizam gráficos, lidam com importação/exportação de dados e implementam recursos como o auto salvamento, mantendo a interface responsiva. Em ambos os casos, as threads permitem a execução de tarefas demoradas sem interromper o usuário.

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

O uso de threads melhora o desempenho de aplicações em ambientes com múltiplos processadores ao permitir a execução paralela de tarefas, distribuir cargas de trabalho em núcleos diferentes e manter a responsividade. Isso é especialmente útil em tarefas intensivas em CPU, como processamento de imagem e simulações. No entanto, o gerenciamento cuidadoso das threads é essencial para evitar problemas de sincronização e concorrência. Em resumo, as threads permitem explorar o potencial de processamento paralelo, acelerando a execução de tarefas em sistemas com vários núcleos.

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

O uso de threads em ambientes cliente-servidor proporciona benefícios como concorrência para atender vários clientes simultaneamente, escalabilidade para acomodar um grande número de clientes, responsividade ao continuar atendendo outras solicitações enquanto aguarda E/S lenta e isolamento de clientes para evitar que problemas em um cliente afetem outros. As threads também facilitam o gerenciamento de conexões, possibilitam a programação paralela e aproveitam múltiplos núcleos de CPU. No entanto, a sincronização e o dimensionamento adequados são essenciais para evitar problemas de concorrência e limitações de escalabilidade.

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

O uso de threads em arquiteturas microkernel é benéfico devido à modularidade e à separação de componentes. As threads permitem isolar módulos, gerenciar tarefas de forma eficiente, facilitar a comunicação entre componentes, economizar recursos, explorar paralelismo interno e manter a flexibilidade do sistema. No entanto, o gerenciamento cuidadoso é necessário para evitar problemas de concorrência e garantir a estabilidade do sistema.

**Questionário 7**

**Defina o que é uma aplicação concorrente e dê um exemplo de sua utilização.**

Uma aplicação concorrente é um programa ou sistema em que diferentes partes podem ser executadas simultaneamente. Um exemplo prático ocorre em sistemas de banco de dados online, onde a concorrência permite transações simultâneas, acesso concorrente aos dados e processamento assíncrono de operações. Isso melhora a eficiência do sistema, aumentando a capacidade de resposta e permitindo que várias operações ocorram ao mesmo tempo, proporcionando uma experiência do usuário mais eficiente. Contudo, é necessário implementar mecanismos de controle para garantir a consistência dos dados e evitar problemas como condições de corrida.

**Considere uma aplicação que utilize uma matriz na memória principal para a comunicação entre vários processos concorrentes. Que tipo de problema pode ocorrer quando dois ou mais processos acessam uma mesma posição da matriz?**

Quando dois ou mais processos concorrentes acessam simultaneamente a mesma posição em uma matriz na memória principal, podem surgir problemas como condições de corrida, competição por recursos e leitura/sobreposição não atômica. Esses problemas podem levar a resultados inconsistentes e imprevisíveis. A solução envolve o uso de mecanismos de sincronização, como semáforos ou mutex, para coordenar o acesso concorrente e garantir a consistência dos dados.

**Explique o que é exclusão mútua e como é implementada.**

Exclusão mútua é um conceito crucial em sistemas concorrentes, garantindo que apenas um processo por vez tenha acesso a uma região crítica ou recurso compartilhado para evitar condições de corrida. Pode ser implementada através de mecanismos como semáforos ou mutex. Semáforos utilizam operações "wait" e "signal" para controlar o acesso, enquanto mutex, ou semáforo binário, possui estados de bloqueado e desbloqueado. Ambos os mecanismos visam garantir a consistência dos dados, permitindo que apenas um processo acesse a região crítica por vez.

**Explique é starvation e como podemos solucionar esse problema.**

Starvation ocorre quando um processo não consegue acessar recursos, apesar de estar pronto para executar. Isso pode ser causado por implementações inadequadas de exclusão mútua em ambientes concorrentes. Para solucionar, estratégias como algoritmos Round Robin com prioridades, limitação de tempo de espera, envelhecimento, prioridades dinâmicas e garantia de justiça no escalonamento são utilizadas. Essas abordagens visam evitar que processos fiquem permanentemente bloqueados, garantindo oportunidades equitativas de acesso aos recursos.

**Defina o que é sincronização condicional e dê um exemplo de sua utilização.**

Sincronização condicional é uma técnica em programação concorrente que coordena a execução de processos com base em condições específicas. Um exemplo prático envolve o uso de variáveis de condição, como no caso de um produtor e um consumidor compartilhando um buffer. A sincronização ocorre quando o produtor espera o buffer não estar cheio, e o consumidor espera o buffer não estar vazio. Essa abordagem garante uma execução ordenada e consistente em sistemas concorrentes, evitando problemas como a inanição e promovendo a interação adequada entre os processos.