**Atividade aula 05 - Questionário**

1. Defina o kernel de um SO.

É responsável por conectar o software ao hardware. Desta forma, ele estabelece uma comunicação eficaz entre os recursos do sistema operacional e administra suas funções.

1. Quais as principais atribuições do kernel?

O núcleo ou kernel é o componente central do [sistema operativo](https://pt.wikipedia.org/wiki/Sistema_operativo) da maioria dos [computadores](https://pt.wikipedia.org/wiki/Computadores); ele serve de ponte entre aplicativos e o processamento real de dados feito a nível de hardware. As responsabilidades do núcleo incluem gerenciar os recursos do sistema, a comunicação entre componentes de [hardware](https://pt.wikipedia.org/wiki/Hardware) e [software](https://pt.wikipedia.org/wiki/Software). Geralmente como um componente básico do sistema operativo, um núcleo pode oferecer a [camada de abstração](https://pt.wikipedia.org/wiki/Camada_de_abstra%C3%A7%C3%A3o) de nível mais baixo para os recursos, especialmente [processadores](https://pt.wikipedia.org/wiki/Unidade_central_de_processamento) e dispositivos de [entrada/saída](https://pt.wikipedia.org/wiki/Entrada/sa%C3%ADda) que softwares aplicativos devem controlar para realizar sua função.

1. Defina system calls.

É o mecanismo programático pelo qual um programa de computador solicita um serviço do núcleo do sistema operacional sobre o qual ele está sendo executado. Para cada serviço existe uma *system call* associada, cada SO possui seu conjunto de chamadas específicos.

1. Qual a principal função das system calls? Qual a causa da sua inexistência?

As system calls podem ser entendidas como uma porta de entrada para o acesso ao núcleo do sistema operacional e a seus serviços. O mecanismo de ativação e comunicação entre o programa e o sistema operacional é semelhante ao mecanismo implementado quando um programa chama uma subrotina.

1. Qual a importância dos modos de acesso?

Para que uma aplicação possa executar uma instrução privilegiada, é necessário que no processador seja implementado o mecanismo de proteção conhecido como modos de acesso. Existem basicamente dois modos de acesso implementados pelos processadores: modo de acesso usuário e [modo de acesso núcleo](https://pt.wikipedia.org/wiki/Modo_n%C3%BAcleo). Quando o processador trabalha no modo usuário, uma aplicação só pode executar instruções não privilegiadas, tendo acesso a um número reduzido de instruções, enquanto no modo núcleo ou supervisor a aplicação pode ter acesso ao conjunto total de instruções do processador. O mecanismo de modos de acesso também é uma boa forma de proteger o próprio núcleo do sistema residente na [memória](https://pt.wikipedia.org/wiki/Mem%C3%B3ria_(computador)) principal.

1. O que são instruções privilegiadas e não privilegiadas? Quando elas podem ser executadas?

**Privilegiadas:** Têm o poder de comprometer o sistema, só devem ser executadas pelo sistema operacional ou sob sua supervisão,impedindo, assim, a ocorrência de problemas de segurança e integridade do sistema.

**Não Privilegiadas:** Não oferecem risco ao sistema, podendo ser executada apenas no modo usuário, é executada no espaço de aplicação, modo aonde normalmente as aplicações são executadas.

1. Diferencie modo usuário e modo kernel.

O modo kernel é conhecido como privilegiado por ter acesso completo ao computador, a todas as partes do hardware. Já o modo usuário, que limita as ações para garantir a segurança e não comprometer nenhuma parte do próprio sistema, é mais difícil de um aplicativo violar a privacidade de outro.

1. O kernel de um SO é sempre executado em qual modo de acesso do processador? Explique.

O kernel é executado no modo de acesso completo, sendo privilegiado, tendo acesso a todas as partes do hardware. Como ele é o núcleo do sistema operacional, ele têm total controle sobre tudo o que ocorre no sistema, como um todo.

1. Explique como é um sistema monolítico.

Sistema único, não dividido, que roda em um único processo, uma aplicação de software em que diferentes componentes estão ligados a um único programa dentro de uma única plataforma.

1. Explique como é um sistema em camadas.

Divide o sistema em camadas “sobrepostas”, onde cada módulo oferece um conjunto de funções nas quais podem ser utilizadas por outros módulos. Os módulos podem servir para fazer referência apenas para módulos das camadas inferiores.

1. Explique como é um sistema cliente-servidor.

Há uma divisão do sistema em processo, aonde cada um é responsável por um conjunto de serviços, basicamente temos o cliente que solicita o serviço e o servidor que é basicamente o processo que responde a solicitação. O núcleo (kernel) têm a função de fazer a comunicação entre cliente e servidor.

1. Com suas palavras, defina processo.

A estrutura responsável pela manutenção de todas as informações necessárias à execução de um programa, como conteúdo dos registradores e endereços de memória.

1. Qual a importância do bloco de controle do processo? Quais informações ele deve conter?

No Bloco de controle o SO mantém todas informações do processo, como: identificação, prioridade, estado corrente, recursos alocados, informações sobre o programa em execução.

1. Quais são os elementos básicos que constituem um processo? Comente cada um deles.

**Contexto de hardware**: Constitui-se do conteúdo dos registradores da UCP. Fundamental para a implementação de sistemas de tempo compartilhado

**Contexto de software**: Especifica características do processo que vão influenciar na execução do processo. Define basicamente três grupos de informações sobre o processo: sua identificação, suas quotas e seus privilégios.

**Espaço de endereçamento:** É a área de memória do processo onde o programa será executado, além do espaço para os dados utilizados por ele.

1. Por quais estados um processo pode passar em sistema multiprogramado? Explique cada um deles.

**Execução (running):** um processo é dito no estado de execução quando está sendo processado pela UCP.

**Pronto (ready):** um processo é dito no estado de pronto quando apenas aguarda uma oportunidade para executar, ou seja, espera que o SO aloque a UCP para a sua execução.

**Espera (wait):** um processo é dito no estado de espera enquanto aguarda algum evento externo ou por algum recurso para poder prosseguir seu processamento.

1. Explique o que é a lista de pronto (*ready*) e a lista de espera (*wait*).

**Pronto** (**ready**): O processo está **pronto** e esperando para ser executado pela CPU.

**Espera** (**wait**): O processo está esperando algum evento externo ou por algum recurso para poder prosseguir seu processamento.

1. Explique o que é evento voluntário e evento involuntário.

Mudança de estado por **eventos** do próprio processo (**eventos voluntários**) ou causados pelo sistema operacional (**eventos involuntários**).

1. Qual a classificação que os processos recebem em relação ao seu tipo? Explique cada um deles.

• **Batch**: processos que não necessitam de intervenção do usuário; têm como função

tratar um conjunto de dados de entrada, realizar o processamento dos dados e

produzir um conjunto de dados de saída, como programas de cálculo numérico,

compilações e backups.

• **Interativos**: processos que interagem diretamente com o usuário, ou seja, são tarefas orientadas a entrada/saída geralmente através de uma interface gráfica, e que

precisam de um tempo de resposta rápido do sistema operacional para o tratamento de requisições no contexto do usuário.

• **Daemons**: processos geralmente carregados durante a inicialização do sistema,

que executam continuamente enquanto o sistema estiver ativo, e esperam em segundo plano até que algum serviço seja requerido. Um exemplo desta classe são

1. Apresente uma vantagem e uma desvantagem em utilizar subprocessos.

Vantagem: Subprocesso possui PCB e espaço de endereçamento próprio, mas pode compartilhar quotas com processo pai.

Desvantagem: Término do processo pai elimina respectiva estrutura de subprocessos filhos

1. O diagrama abaixo mostra as mudanças de estado de um processo. Explique os eventos que podem ocorrer para provocar cada mudança.



O processo pode estar pronto (aguardando para ser executado) e voltar a sua execução (processo é executado pela CPU), enquanto o processo está em execução ele pode ser colocado em espera (pode aguardar por algum evento externo ou algum recurso para sua execução) ou pode mudar para o estado pronto que estará próximo de ser executado, e quando ele está em espera pode apenas ter seu próximo estado em pronto e não ter como aguardar para execução.

1. Apresente uma vantagem e uma desvantagem em utilizar *threads*.

Os threads possuem vantagens e desvantagens ao dividir um programa em vários processos. Uma das vantagens é que isso facilita o desenvolvimento, visto que torna possível elaborar e criar o programa em módulos, experimentando-os isoladamente no lugar de escrever em um único bloco de código. No entanto, uma das desvantagens é que com vários threads o trabalho fica mais complexo, justamente por causa da interação que ocorre entre eles

1. Explique qual a principal diferença entre subprocesso e *thread*.

Quando o processo de origem de um thread morre, os threads também morrem, diferente dos subprocessos. Subprocessos não compartilham de áreas de endereçamento como os threads. Os threads gastam mais tempo e recursos de uma CPU que os subprocessos.