UNIDADE II

Sistemas Operacionais

Prof. Me. Michel Fernandes

Critérios de escalonamento

- Tempo de processador/tempo de UCP: tempo de processador ou tempo de UCP é o tempo que um processo leva no estado de execução durante seu processamento.

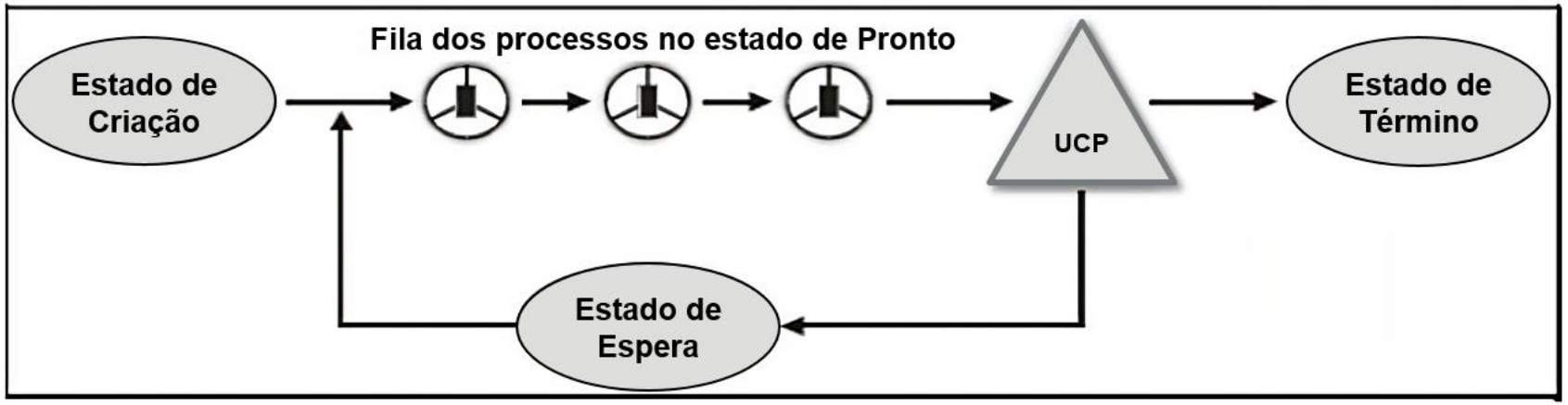
- Tempo de espera: tempo de espera é o tempo total que um processo permanece na fila de pronto durante seu processamento, aguardando para ser executado.

- Tempo de *turnaround*: tempo de *turnaround* é o tempo que um processo leva desde a sua criação até o seu término, levando em consideração todo o tempo gasto na espera para alocação de memória, espera na fila de pronto (tempo de espera), processamento na UCP (tempo de processador) e na fila de espera, como nas operações de E/S.

    - As políticas de escalonamento buscam minimizar o tempo de *turnaround*.

Algoritmos de escalonamento – FIFO – *First In First Out*

- No escalonamento FCFS (*First Come*, *First Served*) ou *first-in-first-out* (FIFO *scheduling*), o processo que chegar primeiro ao estado de pronto é o selecionado para execução. Esse algoritmo é bastante simples, sendo necessária apenas uma fila, em que os processos que passaram para o estado de pronto entram no seu final e são escalonados quando chegam ao seu início.
- Quando o processo em execução termina seu processamento ou vai para o estado de espera, o primeiro processo da fila de pronto é escalonado ou é encerrado.
- Quando saem do estado de espera, todos os processos entram no final da fila de pronto.



Fonte: adaptado de: Machado; Maia (2013, p. 137).

Algoritmos de escalonamento – FIFO – *First In First Out*

- Seu principal problema é a impossibilidade de se prever quando um processo tem sua execução iniciada, já que isso varia em função do tempo de execução dos demais processos posicionados à sua frente na fila de pronto.

- Outro problema nesse tipo de escalonamento é que processos CPU-bound levam vantagem no uso do processador sobre processos I/O-bound. No caso de existirem processos I/O-bound mais importantes do que os CPU-bound, não é possível tratar esse tipo de diferença.

- O escalonamento FIFO é do tipo não preemptivo e foi inicialmente implementado em sistemas monoprogramáveis.

Algoritmos de escalonamento – SJF – tarefa mais curta primeiro

- SJF (*Shortest Job First*): “primeiro a tarefa mais curta”.
- Processo com o tempo esperado de serviço (Ts) menor é selecionado primeiro.
- Favorece processos curtos em detrimento dos longos.
- Dificuldade no uso do algoritmo SJF consiste em estimar *a priori* a duração de cada tarefa, ou seja, antes de sua execução.
    - Possibilidade de inanição (*starvation*) das tarefas mais longas. Caso o fluxo de tarefas curtas chegando ao sistema seja elevado, as tarefas mais longas nunca serão escolhidas para receber o processador e vão literalmente “morrer de fome”, esperando na fila sem poder executar.

Algoritmo de escalonamento – circular

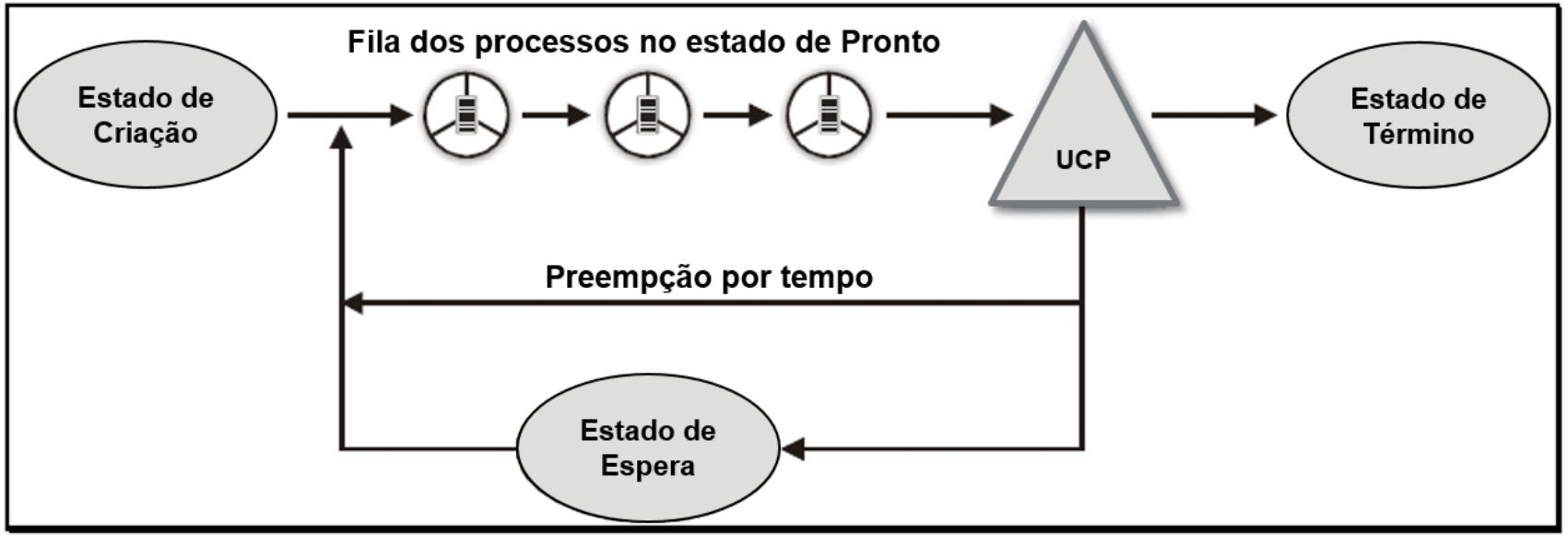
- O escalonamento circular (*round robin scheduling*) é um escalonamento do tipo preemptivo, projetado especialmente para sistemas de tempo compartilhado.

- Esse algoritmo é bastante semelhante ao FIFO; porém, quando um processo passa para o estado de execução, existe um tempo limite para o uso contínuo do processador denominado fatia de tempo (*time-slice*) ou *quantum*.

    - No escalonamento circular, toda vez que um processo é escalonado para execução, uma nova fatia de tempo é concedida. Caso a fatia de tempo expire, o sistema operacional interrompe o processo em execução, salva seu contexto e direciona-o para o final da fila de pronto. Esse mecanismo é conhecido como preempção por tempo.

Algoritmo de escalonamento – circular

- A fila de processos em estado de pronto é tratada como uma fila circular. O escalonamento é realizado alocando a UCP ao primeiro processo da fila de pronto. O processo permanece no estado de execução até que termine seu processamento, voluntariamente passe para o estado de espera, ou que sua fatia de tempo expire, sofrendo, nesse caso, uma preempção pelo sistema operacional.
- Após isso, um novo processo é escalonado com base na política de FIFO.



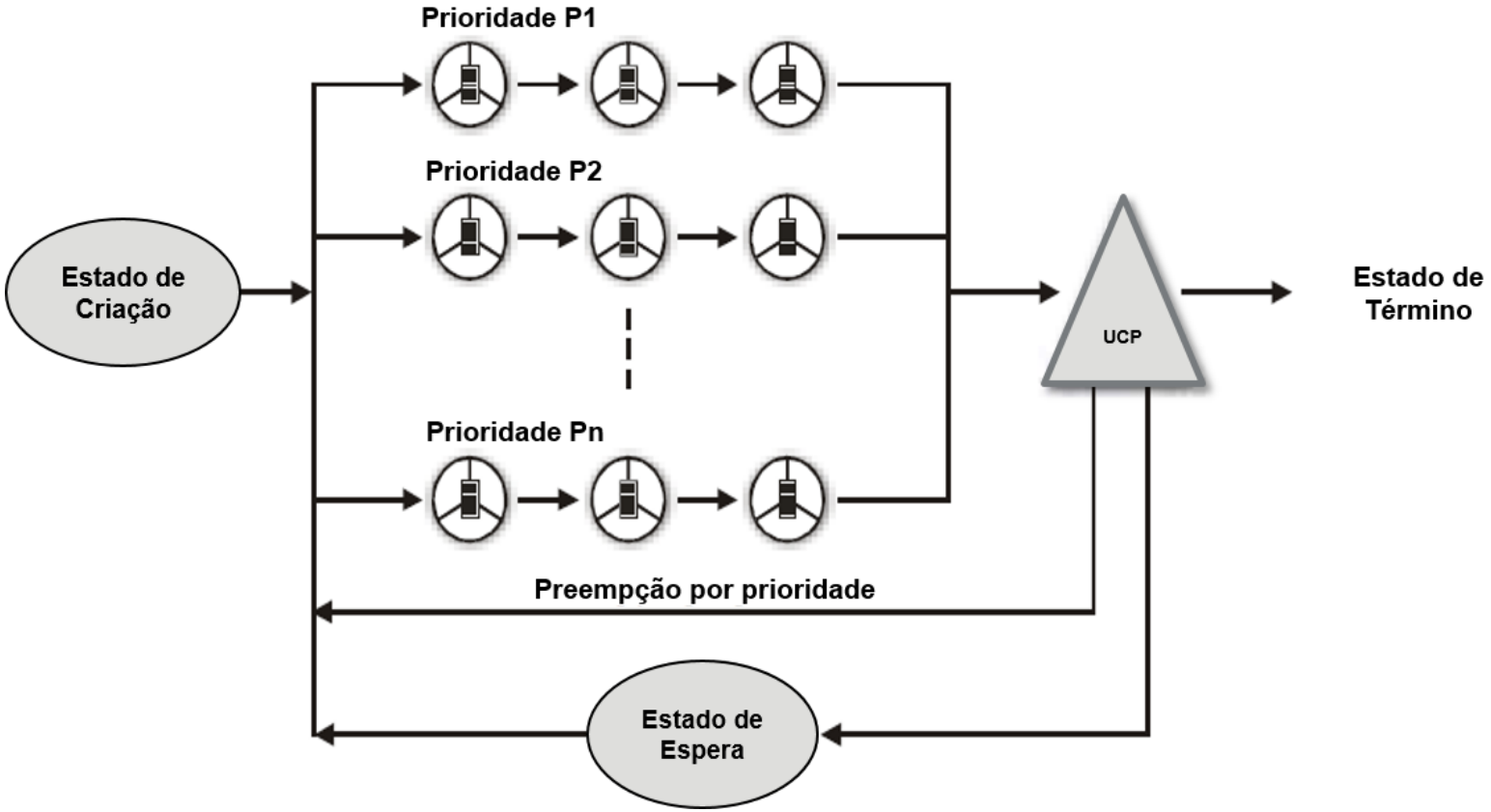
Fonte: adaptado de: Machado; Maia (2013, p. 133).

Algoritmo de escalonamento – por prioridades

- O escalonamento por prioridades é um escalonamento do tipo preemptivo realizado com base em um valor associado a cada processo denominado prioridade de execução.
- O processo com maior prioridade no estado de pronto é sempre o escolhido para execução, e processos com valores iguais são escalonados seguindo o critério de FIFO. Nesse escalonamento, o conceito de fatia de tempo não existe.
- Consequentemente, um processo em execução não pode sofrer preempção por tempo. No escalonamento por prioridades, a perda do uso do processador só ocorre no caso de uma mudança voluntária para o estado de espera ou quando um processo de prioridade maior passa para o estado de pronto.
    - Nesse caso, o sistema operacional deverá interromper o processo corrente, salvar seu contexto e colocá-lo no estado de pronto. Esse mecanismo é conhecido como preempção por prioridade.

Algoritmo de escalonamento – por prioridades

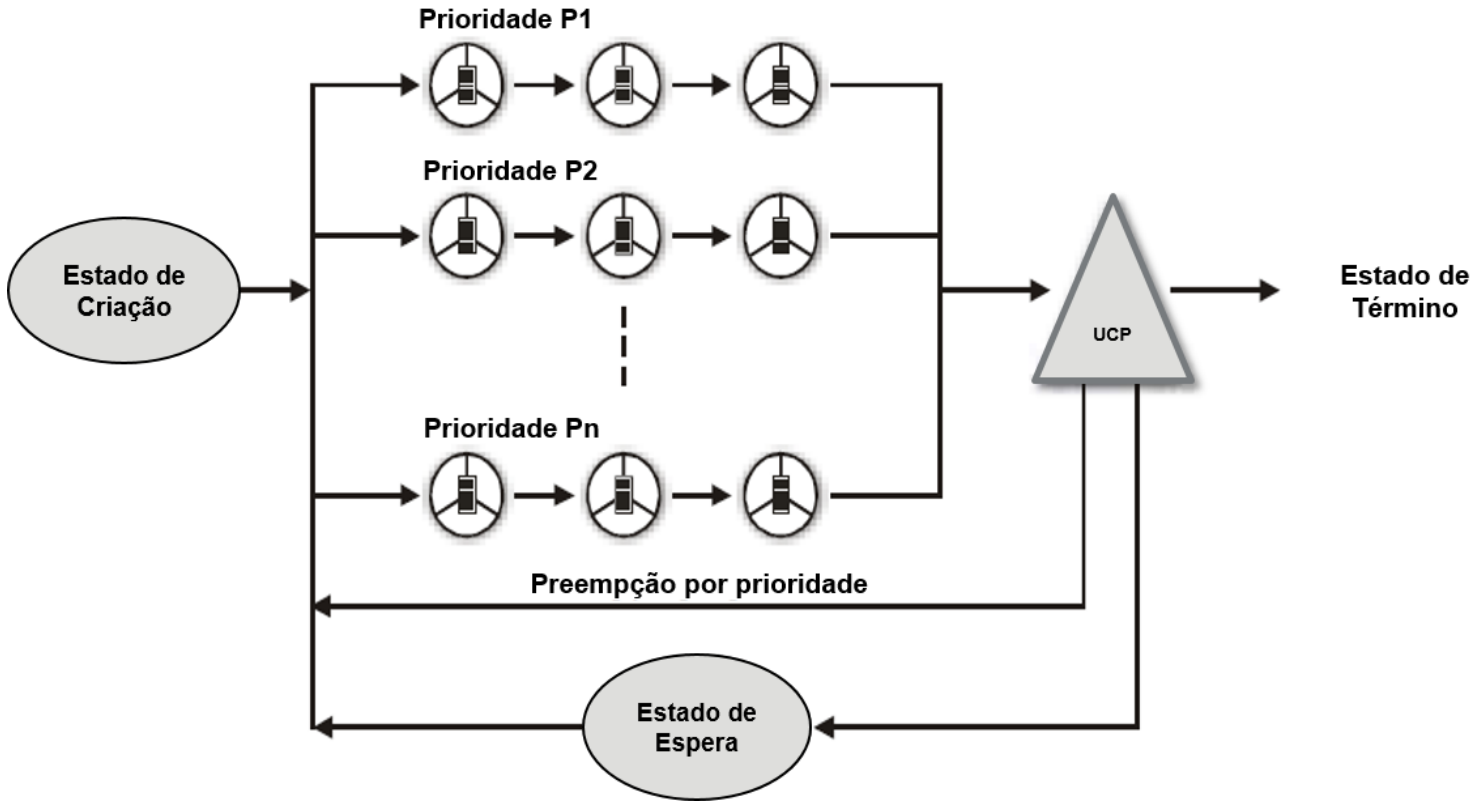
- A fila de processos em estado de pronto é tratada como uma fila circular. O escalonamento é realizado alocando a UCP ao primeiro processo da fila de pronto. O processo permanece no estado de execução até que termine seu processamento, voluntariamente passe para o estado de espera, ou que sua fatia de tempo expire, sofrendo, nesse caso, uma preempção pelo sistema operacional.
- Após isso, um novo processo é escalonado com base na política de FIFO.



Fonte: adaptado de: Machado; Maia (2013, p. 135).

Algoritmo de escalonamento – circular por prioridades

- O escalonamento circular com prioridades implementa o conceito de fatia de tempo e de prioridade de execução associada a cada processo.
- Nesse tipo de escalonamento, um processo permanece no estado de execução até que termine seu processamento, voluntariamente passe para o estado de espera ou saia uma preempção por tempo ou prioridade.
- A principal vantagem é permitir o melhor balanceamento no uso da UCP.



Fonte: adaptado de: Machado; Maia (2013, p. 137).

Interatividade

A partir da lista de pronto, a gerência de processos realiza uma escolha para determinar o próximo processo a ser executado. Qual o nome dessa atividade?

a) Região crítica.
b) Impasse.
c) Thread.
d) Semáforo.
e) Escalonamento.

Resposta

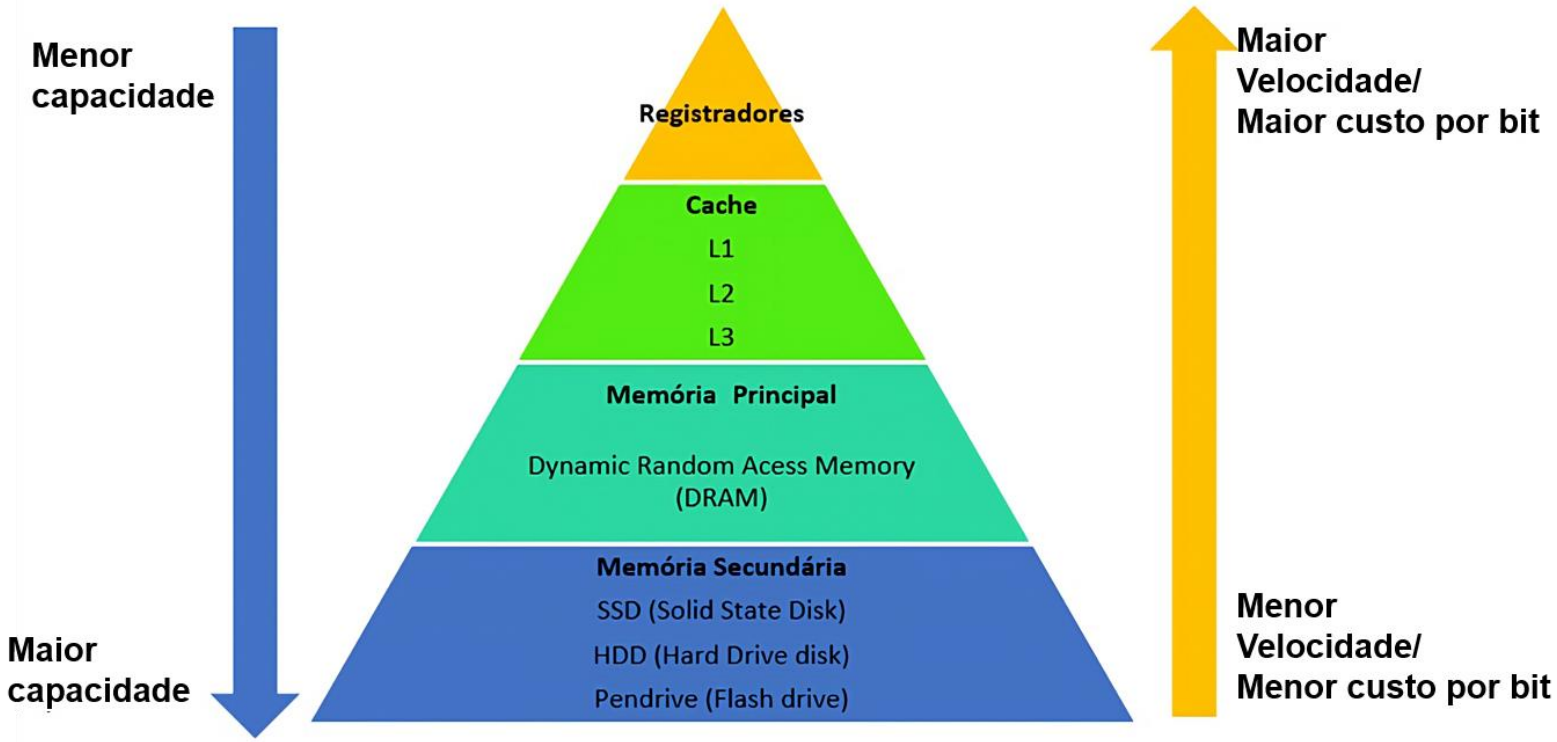
A partir da lista de pronto, a gerência de processos realiza uma escolha para determinar o próximo processo a ser executado. Qual o nome dessa atividade?

a) Região crítica.
b) Impasse.
c) Thread.
d) Semáforo.
e) Escalonamento.

Hierarquia de memória

- Subsistema de memória, projetado de modo que componentes sejam organizados hierarquicamente.

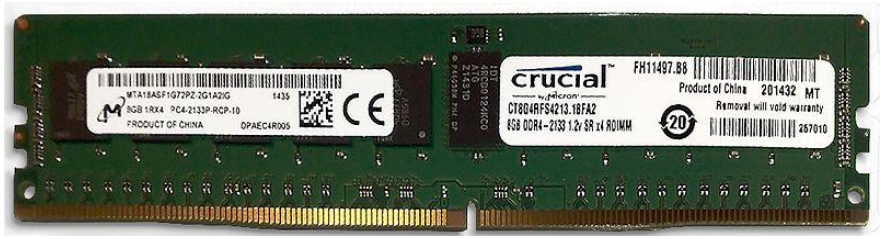
- As características analisadas são: tempo de acesso, capacidade, volatilidade, tecnologia de fabricação, temporariedade e custo.



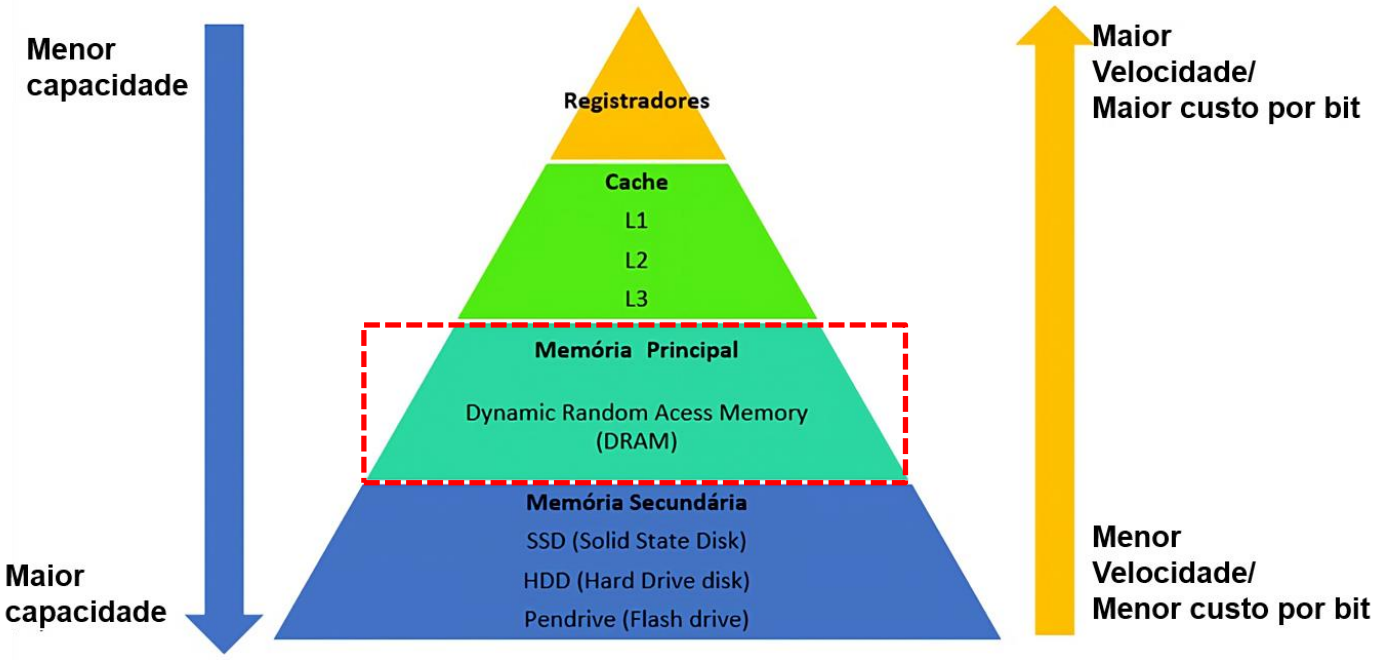
Fonte: autoria própria.

Memória principal

- Memória principal também chamada de memória RAM – *Random Access Memory*.
- Armazenamento de dados dos programas e os próprios programas.
- Memórias que o processador pode endereçar diretamente.
- Acesso aleatório significa que consegue acessar qualquer célula que pode ser alcançada com uma única instrução.



Fonte: RAM module SDRAM 8GB DDR4. Disponível em: https://commons.wikimedia.org/wiki/File:Two_8_GB_DDR4-2133_ECC_1.2_V_RDIMMs_(straightened).jpg>.



Fonte: autoria própria.

Parâmetros da memória principal

- **Tempo de acesso**: da ordem de nanossegundos.
- **Capacidade**: com endereçamento de 32 bits, a maior quantidade de memória disponível era 4 Gb.
  - Com 64 bits, tamanho máximo de memória aumentou significativamente.

- **Volatilidade**: é volátil.
- **Temporariedade**: para que um programa seja executado é necessário que ele esteja armazenado na memória principal.

Tipos de memória

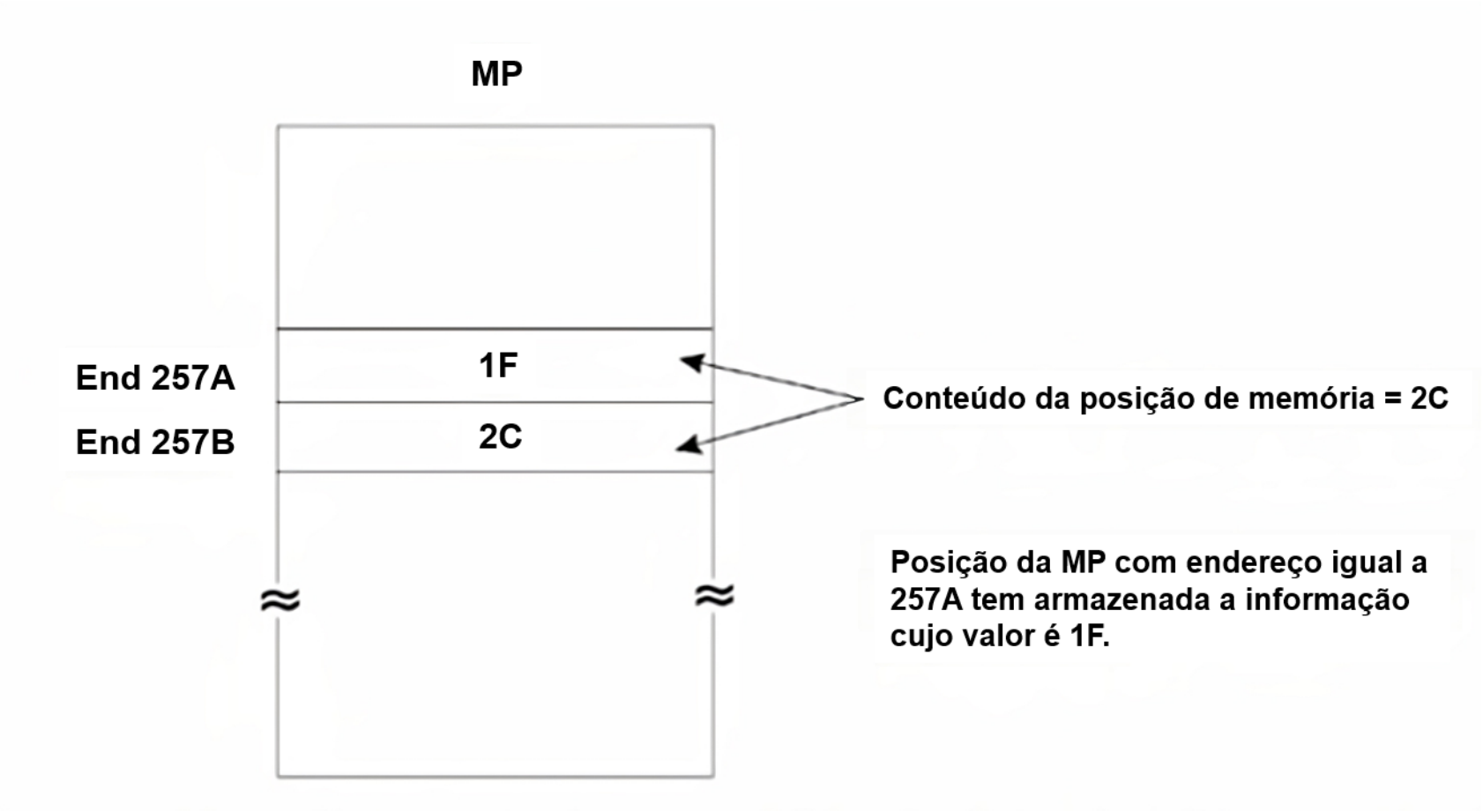
- Principais tipos de memória de semicondutor.
- Memória RAM precisa receber uma fonte de alimentação constante.
- Sem energia, os dados são perdidos.

Tipo de memória	Categoria	Forma de apagar	Mecanismo escrito	Volatilidade
Memória de acesso aleatório (RAM)	Memória de escrita e leitura	Eletricamente, em nível de byte	Eletricamente	Volátil
Memória somente de leitura (ROM)	Memória somente de leitura	Não é possível	Máscara	Não volátil
ROM programável (PROM)			Eletricamente	
PROM apagável (EPROM)	Memória principalmente de leitura	Luz UV, nível de chip		
PROM eletricamente apagável (EEPROM)		Eletricamente, nível de byte		
Memória Flash		Eletricamente, nível de bloco		

Fonte: adaptado de: STALLINGS, William. *Arquitetura e organização de computadores*. 8. ed. São Paulo: Pearson Pratice Hall, 2010, p. 129.

Organização da memória principal

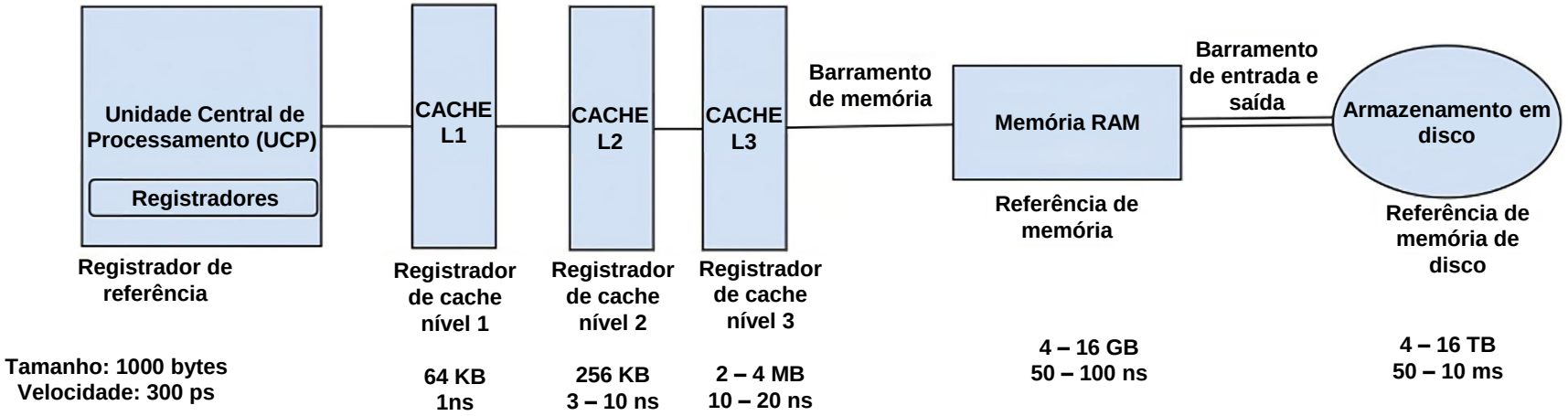
- Palavra: unidade de informação do sistema processador/MP que deve representar o valor de um dado ou uma instrução de máquina.
- Endereço.
- Conteúdo.



Fonte: adaptado de: MONTEIRO, M. A. *Introdução à Organização de Computadores.* 5. ed. Rio de Janeiro: LTC, 2007, p. 96.

Memória Cache

- A velocidade dos processadores é em geral muito maior do que as velocidades das memórias RAM.
- Memória cache soluciona essa limitação entre as velocidades de processamento e a velocidade da memória principal.
- A memória cache é uma memória com menor capacidade que a RAM, porém possui velocidades mais altas.



Fonte: adaptado de: Hennessy (2014).

Conceito de gerenciamento de memória

- Programas são armazenados em memórias secundárias, como discos ou fitas, por ser um meio não volátil, abundante e de baixo custo.

- Como o processador somente executa instruções localizadas na memória principal, o Sistema Operacional deve sempre transferir programas da memória secundária para a memória principal antes deles serem executados.

    - Como o tempo de acesso à memória secundária é muito superior ao tempo de acesso à memória principal, o Sistema Operacional deve buscar reduzir o número de operações de E/S à memória secundária, para evitar problemas de desempenho do sistema.

Funções de gerenciamento de memória

- Deve tentar manter na memória principal o maior número possível de processos residentes.
- Permitir e maximizar o compartilhamento do processador e demais recursos computacionais.
- Mesmo na ausência de espaço livre, o sistema deve permitir que novos processos sejam aceitos e executados.
- Permitir a execução de programas que sejam maiores que a memória física disponível.

Alocação de memória

- Reservar áreas de memória para uso do SO e dos processos.

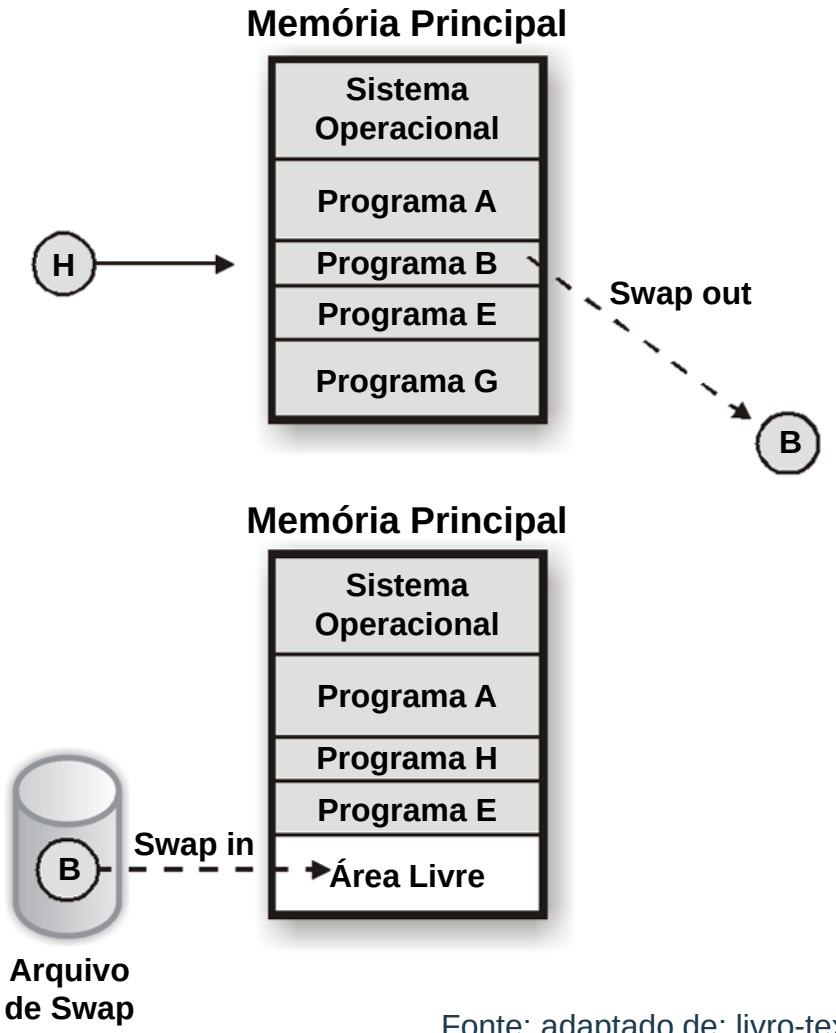
- O alocador de memória:
  - Atende solicitações do núcleo ou de processos.
  - Aloca e libera áreas de memória.
  - Gerencia áreas que estão livres ou ocupadas.
  - Deve ser rápido e eficiente (baixo desperdício de memória).

Swapping ou permuta de memória

- Mesmo na ausência de espaço livre, o sistema deve permitir que novos processos sejam aceitos e executados.

- Isso é possível pela transferência temporária de processos residentes na memória principal para a memória secundária, liberando espaço para novos processos.

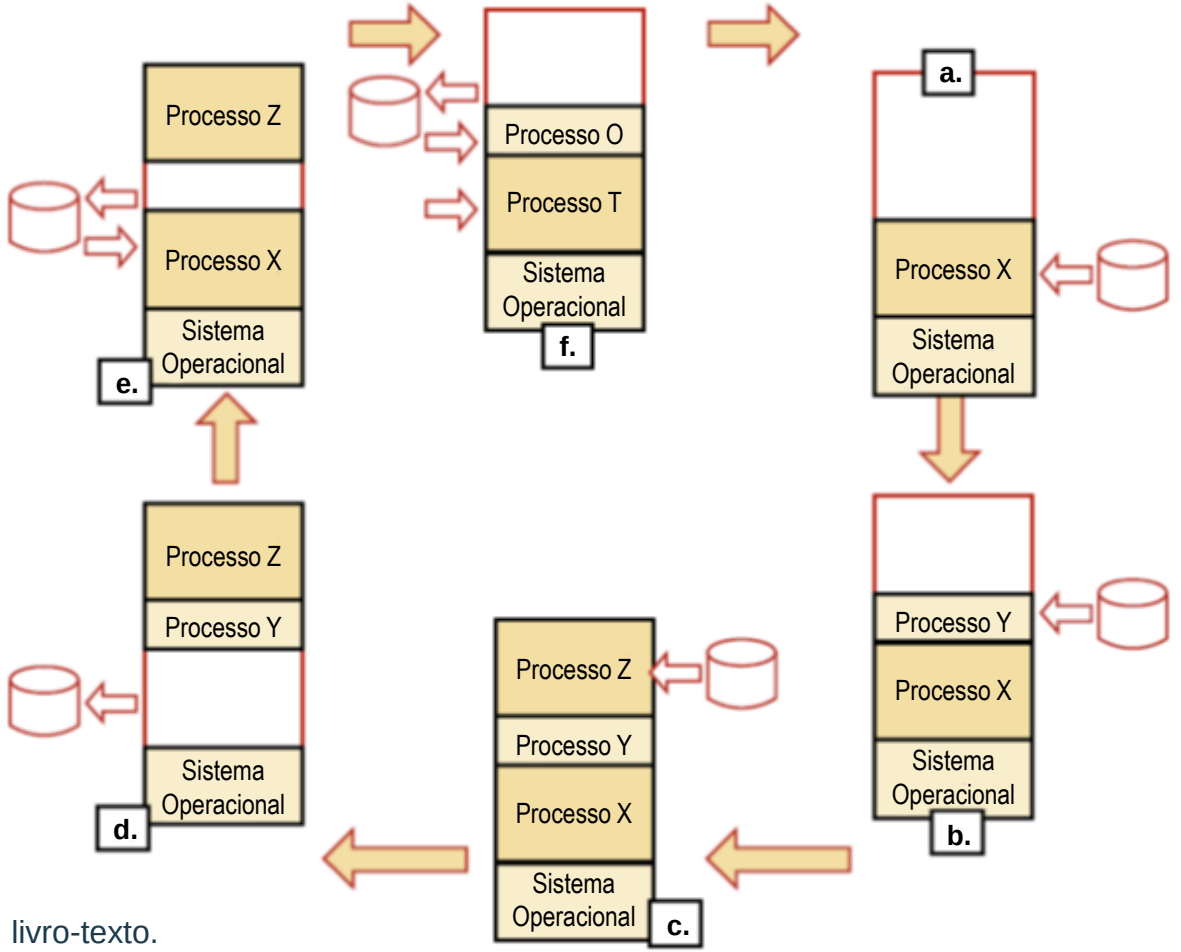
- Esse mecanismo é conhecido como *swapping* ou permuta de memória.



Fonte: adaptado de: livro-texto.

Swapping

- Em a, SO ocupa parte mais baixa da memória e, logo em seguida, temos o processo X ocupando uma parte da memória disponível.
- Em b, um novo processo Y é criado ou trazido do disco para a memória.
- Em c, um novo processo Z é adicionado.
- Em d, o processo X fica ocioso, então, é enviado para o disco rígido.
- Em e/f, outros processos são trocados e o ciclo vai sendo executado até que novos processos entrem e disputem o tempo de CPU e memória.



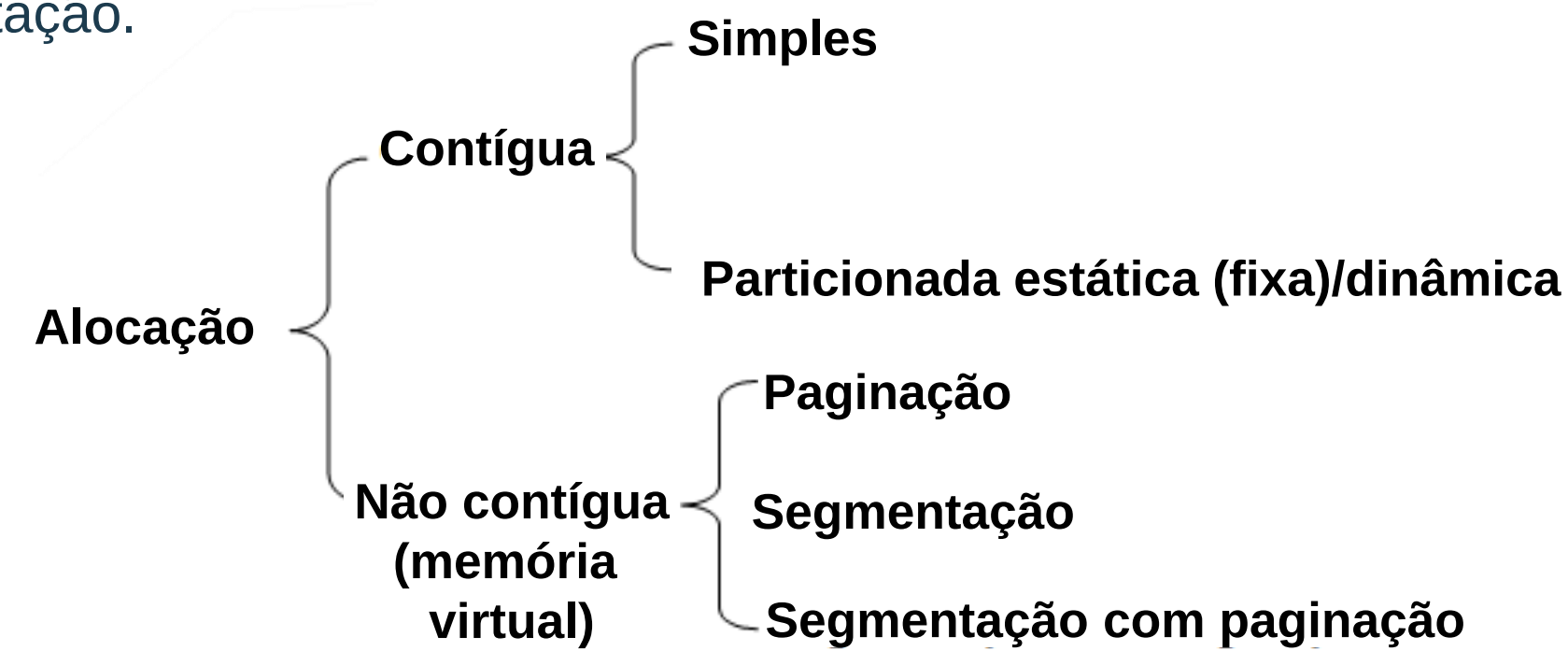
Fonte: adaptado de: livro-texto.

Alocação de memória

- Formas de alocar memória.

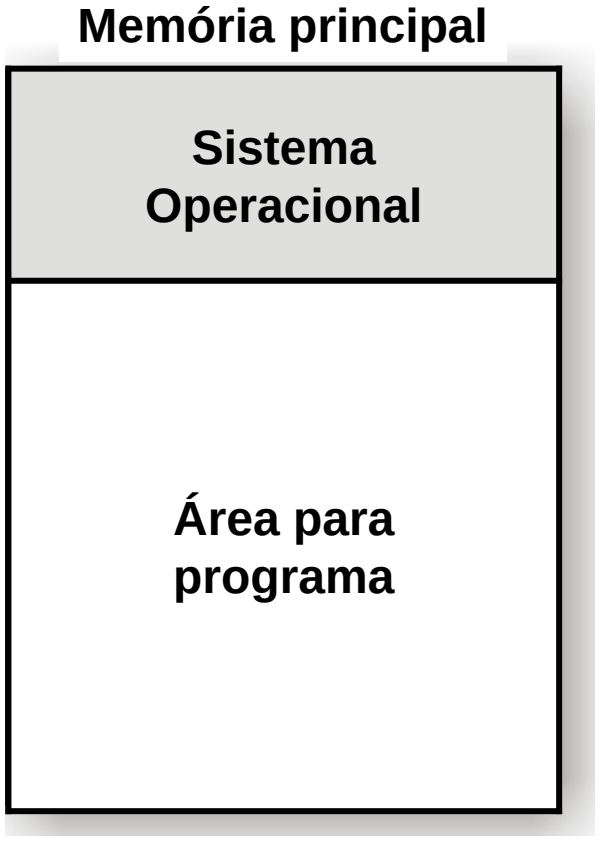
Problemas:

- Gerenciar uma ou mais áreas de memória.
- Atender pedidos de alocação e liberação de blocos.
- Otimizar o uso da memória.
- Evitar/minimizar a fragmentação.



Alocação contígua simples

- Implementada nos primeiros Sistemas Operacionais e para sistemas monoprogramáveis.
- A memória principal é subdividida em duas áreas:
  - Uma para o sistema operacional e outra para o programa do usuário.
  - Dessa forma, o programador deve desenvolver suas aplicações, preocupado em não ultrapassar o espaço de memória disponível.



Alocação contígua simples

- Esquema em que o usuário tem controle sobre toda a memória principal, inclusive a área do Sistema Operacional.

- Implementa controle de proteção do sistema pelo registrador que delimita a área do Sistema Operacional.

- Fácil implementação e código reduzido, porém não utiliza os recursos computacionais de forma eficiente, pois apenas um usuário/aplicação pode dispor desse recurso.

    - Entretanto, somente é aplicável para sistemas monoprogramáveis.

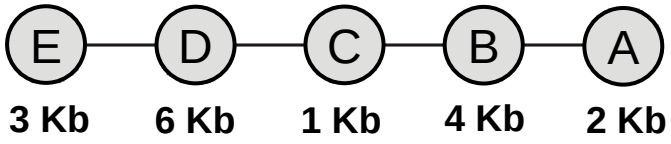
Alocação contígua particionada estática

- Os sistemas multiprogramáveis são muito mais eficientes no uso do processador, necessitando, assim, que diversos programas estejam na memória principal ao mesmo tempo e que novas formas de gerência da memória sejam implementadas.
- Inicialmente, os programas só podiam ser carregados e executados em apenas uma partição específica, mesmo se outras estivessem disponíveis.
- No exemplo anterior, supondo que os programas A e B estivessem sendo executados, os programas C e E não poderiam ser processados na terceira partição, mesmo esta estando livre.

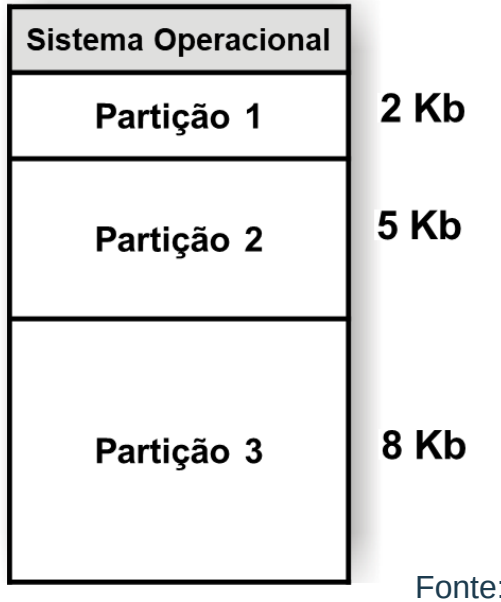
Tabela de partições

Partição	Tamanho
1	2 Kb
2	5 Kb
3	8 Kb

Programas a serem executados:



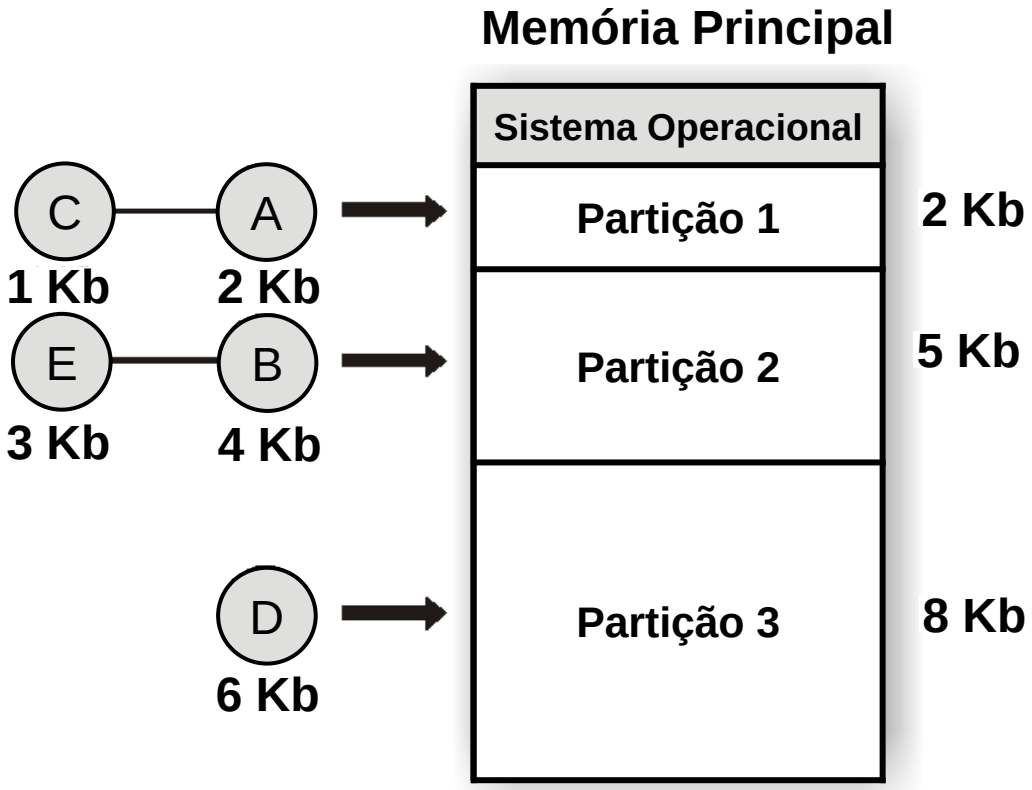
Memória Principal



Fonte: adaptado de: acervo pessoal.

Alocação contígua particionada fixa relocável

- Com a evolução dos compiladores, montadores, ligadores e carregadores, o código gerado deixou de ser absoluto e passou a ser relocável.
- No código relocável, todas as referências a endereços no programa são relativas ao início do código e não a endereços físicos de memória.
- Desta forma, os programas puderam ser executados a partir de qualquer partição.

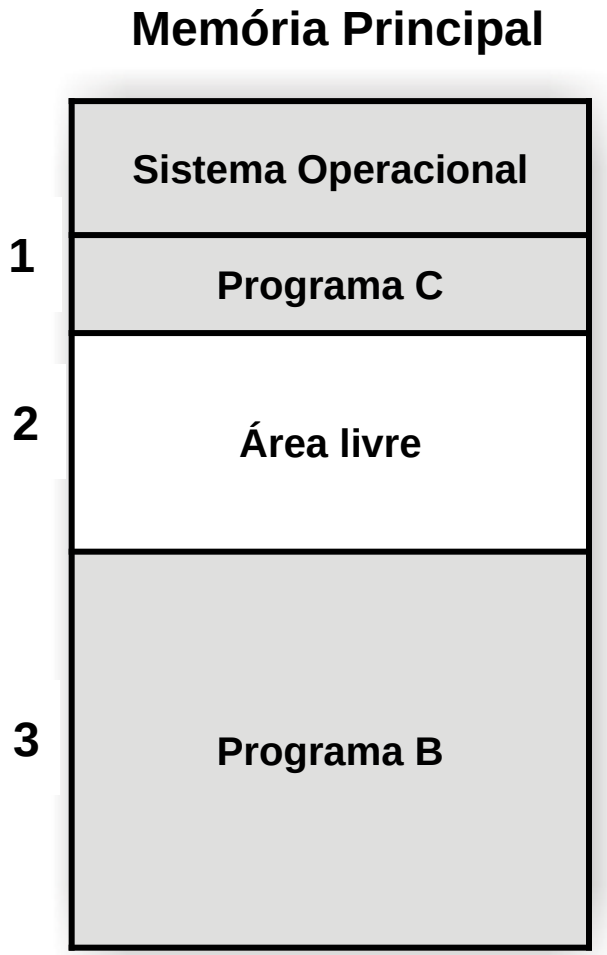


Fonte: adaptado de: acervo pessoal.

Alocação contígua particionada fixa relocável

- Para manter o controle sobre as partições alocadas, a gerência de memória mantém uma tabela com o endereço inicial de cada partição, seu tamanho e se está em uso ou não.

Partição	Tamanho	Livre
1	2 Kb	Não
2	5 Kb	Sim
3	8 Kb	Não



Fonte: adaptado de: acervo pessoal.

Fragmentação interna

- Tanto nos sistemas de alocação fixa quanto nos de alocação relocável, os programas, normalmente, não preenchem totalmente as partições onde são carregados, deixando área de memória livre.

- Esse problema é conhecido como fragmentação interna.

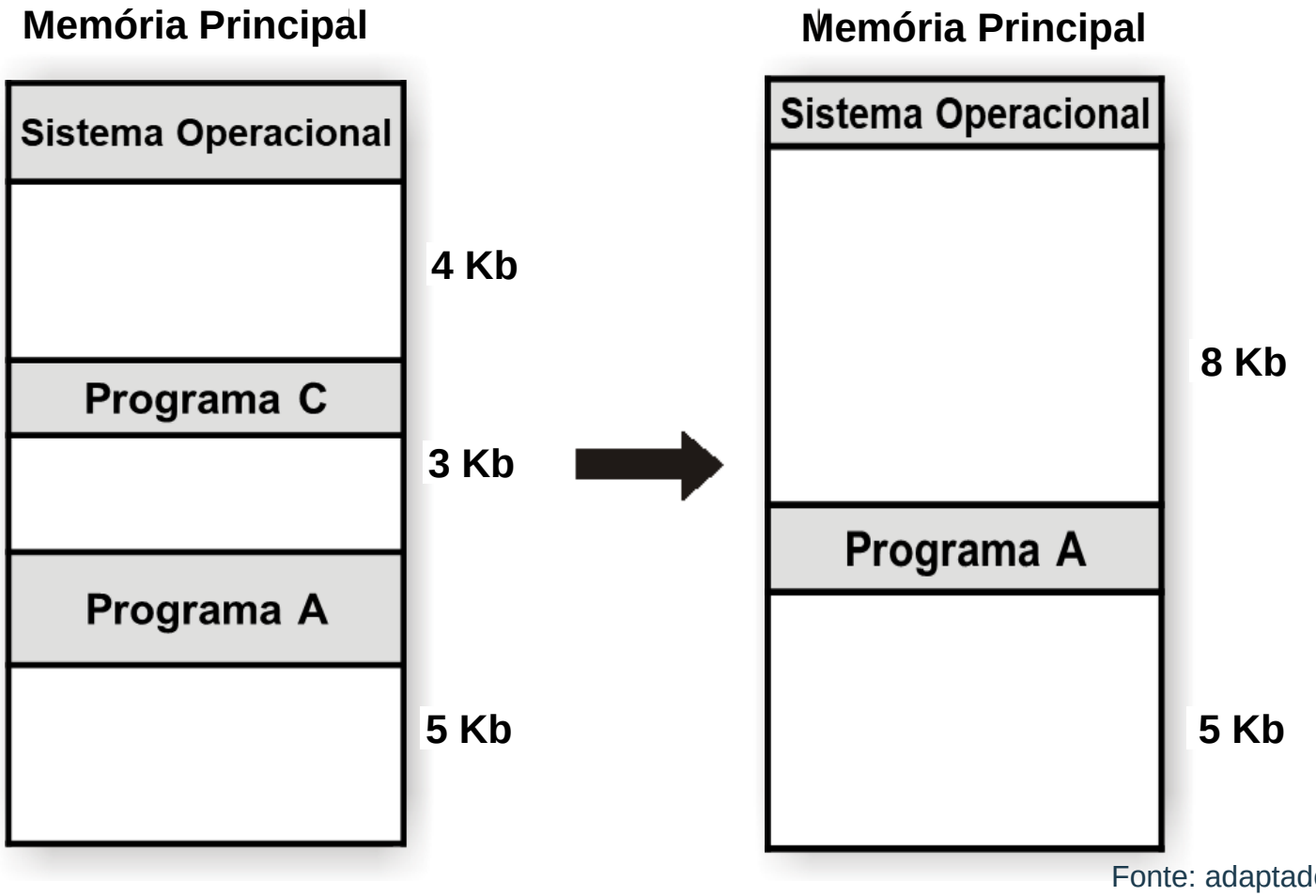
Alocação contígua particionada dinâmica

- Na alocação particionada dinâmica ou variável, foi eliminado o conceito de partições de tamanho fixo. Nesse esquema, cada programa utilizaria o espaço necessário, tornando essa área sua partição.

- Como cada programa utiliza apenas o espaço que necessita, o programa de fragmentação interna não ocorre.

Fragmentação externa

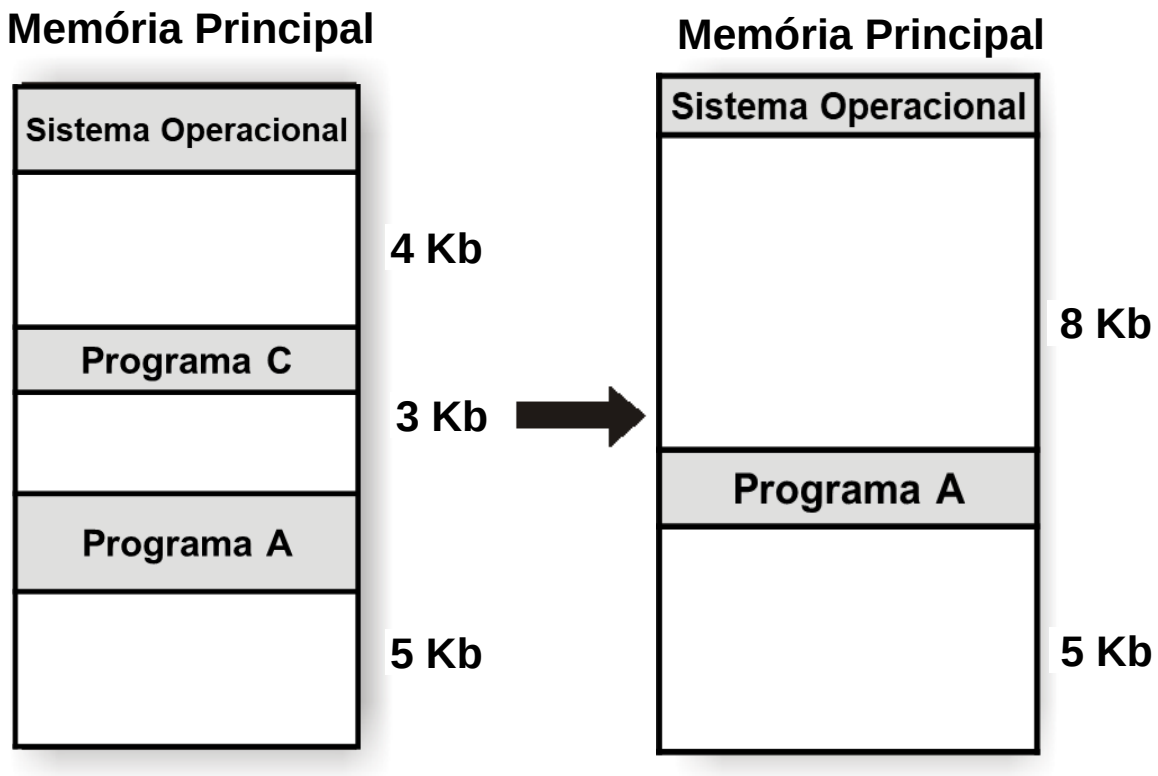
- Os programas foram terminando sua execução e saindo da memória principal e deixavam espaços cada vez menores na memória, não permitindo o ingresso de novos programas que fossem maiores que o bloco deixado.



Fonte: adaptado de: acervo pessoal.

Soluções para o problema de fragmentação externa

- Conforme os programas terminem, apenas os espaços livres adjacentes sejam reunidos, produzindo áreas livres de tamanho maior.

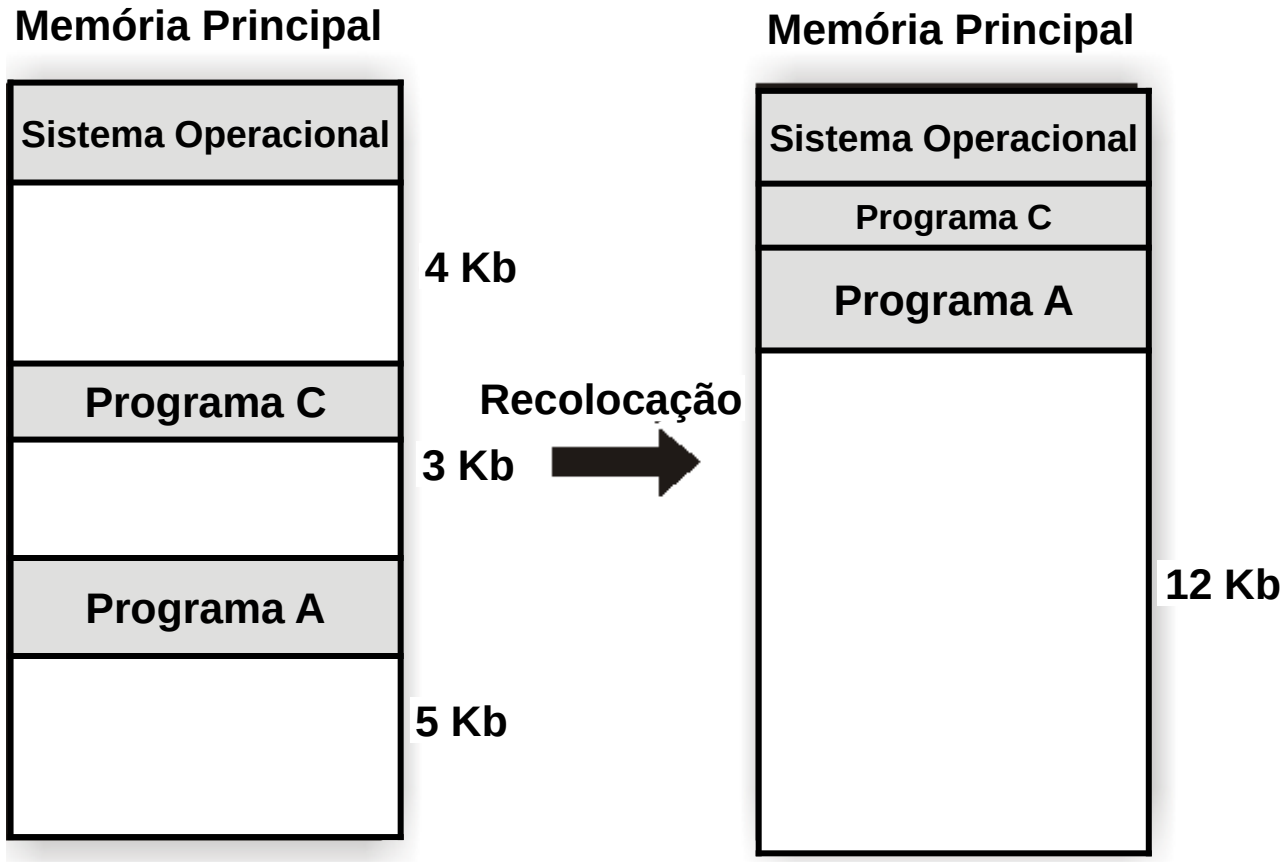


- Relocação de todas as partições ocupadas, eliminando todos os espaços entre elas e criando uma única área livre contígua.

  - Para que esta solução possa ser implementada, é necessário que o sistema tenha a capacidade de mover os diversos programas na memória principal, ou seja, realizar a relocação dinâmica.

Soluções para o problema de fragmentação externa

- Relocação de todas as partições ocupadas, eliminando todos os espaços entre elas e criando uma única área livre contígua.
- Para que esta solução possa ser implementada, é necessário que o sistema tenha a capacidade de mover os diversos programas na memória principal, ou seja, realizar a relocação dinâmica.



Alocação contígua particionada dinâmica

Vantagens:

- Reduz em muito o problema da fragmentação.

Desvantagem:

- Aumenta a complexidade do algoritmo e o consumo de recursos do sistema (processador e área de disco).

Algoritmos de alocação

- Sistema operacional deve decidir qual bloco livre será associado a um processo.

- Algoritmos de alocação.
- Best-fit.
- First-fit.
- Next-fit.
- Worst-fit.

Algoritmos de alocação – First-fit

- “O que primeiro couber.”
- Busca espaço livre desde o início da memória.
- Escolhe o primeiro bloco disponível que seja grande o suficiente.
- Método tenta primeiro utilizar as áreas livres de endereços mais baixos.
- Boa chance de se obter uma grande partição livre nos endereços mais altos.
- Algoritmo mais rápido dos três (*best / worst / first*).
- A lista de áreas livres está ordenada por endereços crescentemente.
- Consome menos recursos para a busca.

Algoritmos de alocação – Best-fit

- “O que melhor couber.”
- Escolhe-se a partição onde o processo deixa o menor espaço sem utilização.
- O objetivo é garantir melhor escolha de partição livre.
- Desvantagem do algoritmo.
  - Escolha da partição mais aproximada resulta em pequenas partições livres.
  - A tendência é ter grande quantidade de pequenas áreas livres não contíguas.
  - Aumentando o problema da fragmentação.
- Solução pode ser o emprego de compactação de memória.

Algoritmos de alocação – Worst-fit

- Escolhe-se a partição onde o processo deixa o maior espaço sem utilização.
- Escolhe o maior espaço livre na memória.
- Nesse algoritmo, a lista de áreas livres deve estar ordenada por tamanho para otimizar busca.
- Comparado ao best-fit, reduz (não elimina) o problema da fragmentação.

Algoritmos de alocação – Next-fit

- Similar ao first-fit.

- Diferença está na busca, que ocorre a partir do endereço da última posição alocada, e não a primeira posição de memória.

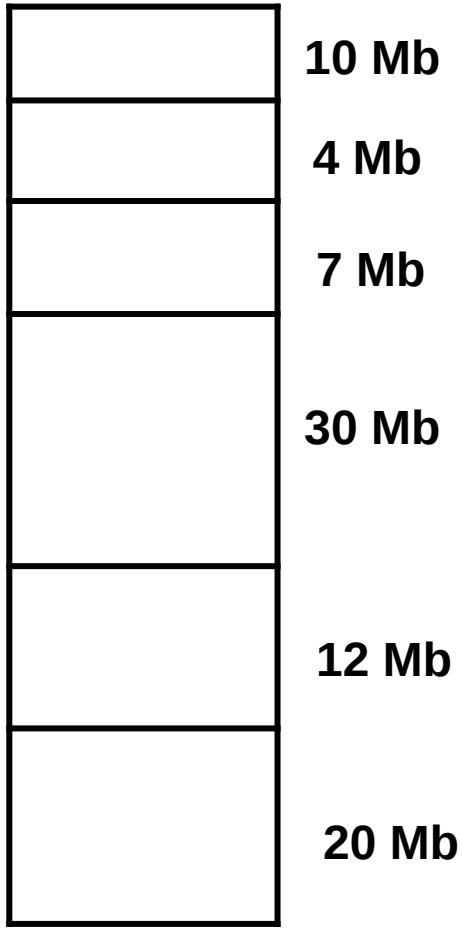
Interatividade

Considere os espaço de memórias livres da figura.

O próximo processo a ser alocado irá ocupar 15 Mb de memória.

Em qual lacuna esse processo será alocado?

a) 10 Mb.
b) 7 Mb.
c) 30 Mb.
d) 12 Mb.
e) 20 Mb.



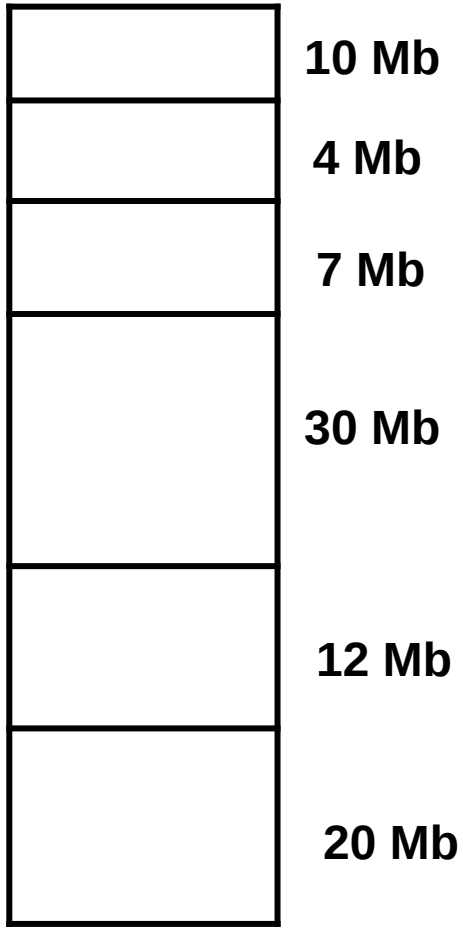
Resposta

Considere os espaço de memórias livres da figura.

O próximo processo a ser alocado irá ocupar 15 Mb de memória.

Em qual lacuna esse processo será alocado?

a) 10 Mb.
b) 7 Mb.
c) 30 Mb.
d) 12 Mb.
e) 20 Mb.



Gerência de memória virtual

- Técnica sofisticada de gerência de memória em que a memória principal, a memória secundária e o processador são combinados, dando a ilusão de existir uma memória maior que a MP real.

- Espaço de endereçamento virtual é aquele que representa um conjunto de endereços virtuais, ou seja, espaços reservados no HD ou processador.

- Espaço de endereçamento real é aquele que representa um conjunto de endereços reais, ou seja, espaços físicos existentes na memória principal.

Memória virtual (MV)

- Os programas podem fazer referência a endereços virtuais fora da memória principal.
- Memória secundária como extensão da MP.
- Na execução de um programa, apenas parte do código fica na memória principal, sendo que seu complemento fica na memória secundária até ser referenciado.
- Aumenta o compartilhamento da memória principal.

Mapeamento

- É o mecanismo que transforma os endereços virtuais em reais.
- Traduz um endereço localizado no espaço virtual para um associado no espaço real.
- Depois de traduzido, o endereço real pode ser utilizado pelo processador para acesso à memória principal.

Alocação de memória virtual por paginação

- Técnica em que o espaço de endereçamento virtual e o espaço de endereço real são divididos em blocos de mesmo tamanho, chamados Páginas.

- Cada processo possui sua própria tabela de páginas, e cada página virtual possui uma ETP (Entrada na Tabela de Páginas), com informações do mapeamento que localiza a página real correspondente.

- Um dos campos principais da ETP é o Bit de Validade, que indica se uma página está ou não na MP. Se o bit for 0, não está carregada (page fault). Nesse caso, o sistema transfere a página da memória secundária para a memória principal, realizando page in, ou paginação.

    - Se o bit for igual a 1, a página está carregada na MP.

Alocação de memória virtual por paginação

- Na política de substituição de páginas, o SO seleciona qual página será liberada da memória principal para que outra a substitua.

É dividida em:

- Política de Substituição de Páginas Local. Sempre que um processo necessitar de uma nova página, o sistema selecionará uma página do processo em questão para ser substituída.
- Política de Substituição de Páginas Global. Todas as páginas alocadas na memória principal são candidatas à substituição, independentemente do processo que a gerou.

Alocação de memória virtual por paginação

- Na política de alocação páginas, o SO seleciona qual página será liberada da memória principal para que outra a substitua.
- Define quantos frames cada processo poderá manter na memória principal.

Há duas alternativas:

- Alocação fixa – Cada processo tem um número máximo de frames que pode ser utilizado na execução do programa. Caso o número de páginas reais seja insuficiente, uma página do processo em questão é descartada para que uma nova página seja carregada.
    - Alocação variável – O número máximo de páginas alocadas varia dependendo da taxa de paginação e da ocupação da memória principal.

Alocação de memória virtual por paginação

Na Política de Busca de Páginas, é definido quando uma página deve ser carregada para a memória principal. Existem duas estratégias:

- Paginação por demanda – Em que as páginas dos processos são transferidas da memória secundária para a memória principal apenas quando são referenciadas. Leva somente as páginas necessárias à execução do programa.

- Paginação antecipada – Carrega para a memória principal, além da página referenciada, outras páginas que podem ou não ser necessárias ao processo ao longo do processamento.

Alocação de memória virtual por segmentação

- Divide o espaço de endereçamento virtual, é dividido em blocos de tamanhos diferentes chamados de segmentos.

- Oferece flexibilidade, permitindo que o tamanho do segmento seja alterável.

- Causa fragmentação externa: em que há várias áreas livres na memória principal, mas nenhuma é suficiente para alocar um novo segmento.

Interatividade

A técnica chamada paginação é usada na maioria dos sistemas de memória virtual.

A memória virtual é dividida em unidades de espaçamento de endereços adjacentes chamadas de páginas. Qual o termo utilizado para essas unidades das memórias?

a) Segmentação.
b) Memória principal (RAM).
c) Paginação.
d) Memória virtual.
e) Frames.

Resposta

A técnica chamada paginação é usada na maioria dos sistemas de memória virtual.

A memória virtual é dividida em unidades de espaçamento de endereços adjacentes chamadas de páginas. Qual o termo utilizado para essas unidades das memórias?

a) Segmentação.

b) Memória principal (RAM).

c) Paginação.

d) Memória virtual.

e) Frames.

Sistema de arquivos

- Sistemas de arquivos
- Os arquivos são gerenciados pelo SO de maneira a facilitar o acesso dos usuários ao seu conteúdo. A parte do SO responsável por essa gerência é o sistema de arquivos, que é a parte mais visível do SO, pois a manipulação de arquivos é frequente.

- Arquivos
- São constituídos por informações logicamente relacionadas e podem representar instruções ou dados.

Atributos de arquivos

- Nome.
- Localização.
- Tipo de arquivo.
- Tamanho.
- Datas de criação / último acesso / modificação.
- Permissões de acesso atribuídas a cada usuário (leitura, escrita, execução, remoção).

Organização de arquivos

- Consiste em como os dados são internamente armazenados, e sua estrutura pode variar de acordo com o tipo de informação contida no arquivo.
- A forma mais simples de organizar é por meio de uma sequência não estruturada de bytes, em que o sistema de arquivos não impõe nenhuma estrutura lógica para os dados.
- A aplicação define toda a organização e critérios.
- Vantagens: flexibilidade para criar diferentes estruturas de dados.
    - As organizações mais conhecidas e implementadas são: a sequencial e a indexada.

Métodos de acesso a arquivos

- Dependendo de como o arquivo está organizado, o sistema de arquivos poderá recuperá-lo de diferentes maneiras:

- Acesso sequencial – Utilizado para arquivos em fitas magnéticas, e o acesso era restrito à leitura dos registros na ordem em que eram gravados, e os novos registros eram gravados no final de cada arquivo.

- Acesso direto – Surgiu com os discos magnéticos, permitia a leitura/gravação de um arquivo diretamente na sua posição, era realizado por meio do número do registro.

    - Acesso indexado ou acesso por chave – Tem como base o acesso direto. O arquivo deve possuir uma área de índice em que existam ponteiros para os diversos registros. Caso a aplicação deseje acessar um registro, deve especificar uma chave por meio da qual o sistema pesquisará na área de índice o ponteiro correspondente, e assim é realizado um acesso direto ao registro desejado.

Operações de entrada e saída

- O sistema de arquivos disponibiliza um conjunto de rotinas que permitem as aplicações realizarem operações de E/S, como tradução de nomes em endereços, leitura e gravação de dados, criação e eliminação de arquivos.
- As rotinas de E/S tem como função disponibilizar uma interface simples e uniforme entre a aplicação e os diversos dispositivos.

ROTINA	DESCRIÇÃO	ATRIBUTOS
CREATE	Criação de arquivos	São informações de controle de cada arquivo. Variam dependendo do sistema de arquivos, porém, estes estão presentes em quase todos os sistemas: tamanho, proteção, identificação do criador, data da criação.
OPEN	Abertura de um arquivo	
READ	Leitura de um arquivo	
WRITE	Gravação de um arquivo	
CLOSE	Fechamento de um arquivo	
DELETE	Eliminação de uma arquivo	

Estrutura de diretório

- É como o sistema de arquivos organiza logicamente os diversos arquivos contidos em um disco, onde armazena informações como: localização física etc.

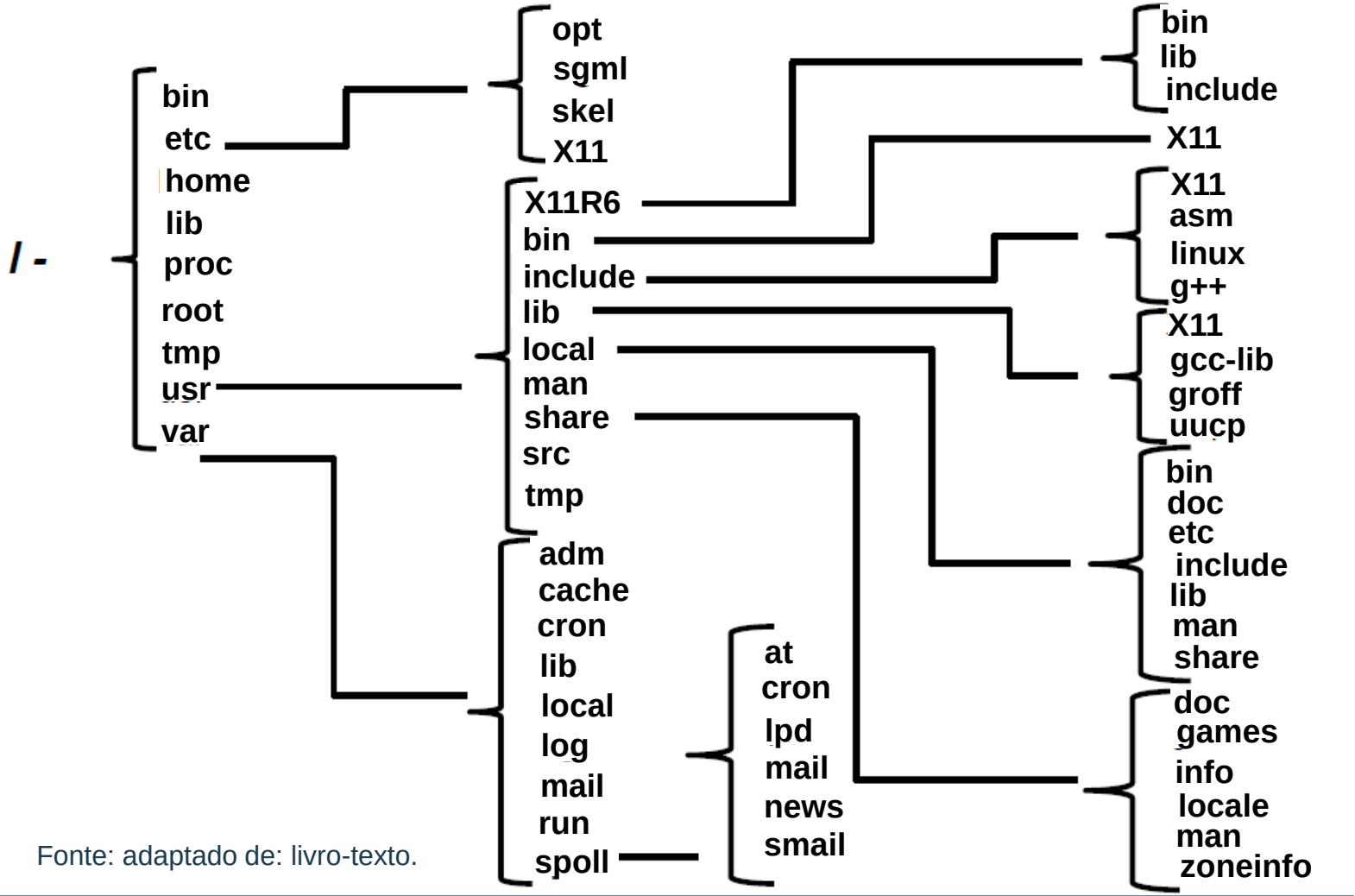
- Quando um arquivo é aberto, o SO procura sua entrada na estrutura de diretórios, armazenando as informações sobre os atributos e a localização de um arquivo em uma tabela mantida na memória principal.

- A implementação mais simples de uma estrutura de diretórios é chamada de Nível Único, em que só existe um único diretório contendo todos os arquivos do disco.

    - É bastante limitado, pois não permite que os usuários criem arquivos com o mesmo nome, o que ocasionaria conflito no acesso aos arquivos.

Estrutura de diretório

- A figura representa uma parte da árvore de diretórios típica de um sistema Linux, cuja estrutura é definida nas normas *Filesystem Hierarchy*.



Fonte: adaptado de: livro-texto.

Gerência de espaço livre em disco

- A forma mais simples de implementar uma estrutura de espaços livres é por meio de uma tabela denominada Mapa de Bits (BITMAP). Cada entrada na tabela é associada a um bloco do disco representado por um bit, podendo assumir valor igual a 0 (bloco livre) ou 1 (bloco ocupado).

- Uma segunda maneira é encadear todos os blocos livres do disco, em que cada bloco possui uma área para armazenar o endereço do próximo bloco.

  - Outra solução leva em consideração que blocos contíguos são geralmente alocados e liberados simultaneamente, mantendo uma tabela com o endereço do 1º bloco e o número de blocos livres contíguos que seguem.

Gerência de alocação de espaço livre em disco

- Alocação contígua: Armazena um arquivo em blocos sequenciais dispostos no disco. O sistema localiza um arquivo por meio do endereço do 1º bloco.

- Seu principal problema é a alocação de espaço livre para novos arquivos. Caso um arquivo deva ser criado com um determinado tamanho, é necessário existir uma quantidade suficiente de blocos contíguos no disco para a realização de alocação.

- Existem estratégias de alocação para selecionar qual o segmento será alocado:

- FIRST-FIT – O primeiro segmento livre com tamanho suficiente será alocado.

  - BEST-FIT – Seleciona o menor segmento livre disponível com tamanho suficiente para armazenar o arquivo.

  - WORST-FIT – O maior segmento é alocado. Entretanto, cria problemas de fragmentação do espaço livre.

Interatividade

Podemos definir um arquivo como sendo um conjunto de dados armazenados em um dispositivo físico não volátil, com um nome e/ou referência que permita sua localização futura. Cada arquivo é diferenciado por um conjunto de atributos. Tipicamente, podemos definir os seguintes atributos:

I. Nome: uma sequência de caracteres para identificar de tal forma que um ser humano, ao vê-lo na tela, consiga identificá-lo.

II. Data: de criação do arquivo, de último acesso, de modificação.

III. Tipo: indica se o formato do arquivo é áudio, vídeo, imagem, texto ou outro.

São atributos dos arquivos:

a) I, apenas.

b) I e II, apenas.

c) I e III, apenas.

d) II e III, apenas.

e) I, II e III.

Resposta

Podemos definir um arquivo como sendo um conjunto de dados armazenados em um dispositivo físico não volátil, com um nome e/ou referência que permita sua localização futura. Cada arquivo é diferenciado por um conjunto de atributos. Tipicamente, podemos definir os seguintes atributos:

I. Nome: uma sequência de caracteres para identificar de tal forma que um ser humano, ao vê-lo na tela, consiga identificá-lo.

II. Data: de criação do arquivo, de último acesso, de modificação.

III. Tipo: indica se o formato do arquivo é áudio, vídeo, imagem, texto ou outro.

São atributos dos arquivos:

a) I, apenas.

b) I e II, apenas.

c) I e III, apenas.

d) II e III, apenas.

e) I, II e III.

Referências

- FURUKAWA, F.; NUNES, R. *Fundamentos de sistemas operacionais*. São Paulo: Editora Sol, 2011.
- HENESSY, J. L.; PATTERSON, D. A. *Arquitetura de computadores*: uma abordagem quantitativa. 5. ed. Rio de Janeiro: Campus, 2014.
- MACHADO, F. B., MAIA, L. P. *Arquitetura de sistemas operacionais*. 5. ed. Rio de Janeiro, LTC: 2013.
- MAZIERO, C. A. *Sistemas operacionais*: conceitos e mecanismos. (Recurso eletrônico). Curitiba: DINF/UFPR, 2019.

ATÉ A PRÓXIMA!