

**UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE SISTEMAS DA INFORMAÇÃO**

Pedro Henrique Gomes Magri

Simulador de gerenciamento de memória com paginação

FLORIANÓPOLIS
2024

Sobre o sistema

O projeto simula o gerenciamento de memória utilizando conceitos de alocação de memória não contígua. O programa é capaz de criar processos com tamanhos aleatórios, alocar páginas de memória para esses processos e visualizar a utilização da memória física e as tabelas de páginas dos processos.

Os valores de tamanho de memória física, tamanho de página e tamanho máximo de um processo são configuráveis no programa e identificados como definições dentro do script “memoria.h”

Os arquivos foram divididos com a ideia de otimizar a manutenção e melhor entendimento do código, proporcionando maior independência dos serviços na medida do possível, representado abaixo:

```
|— Makefile
|— README.md
|— includes
|   |— memoria.h  Declarações das funções relacionadas à memória
|   |— pagina.h   Definição da estrutura de Página.
|   |— processo.h Declarações das funções relacionadas e definição da estrutura Processo.
|   |— quadro.h   Declarações das funções relacionadas e definição da estrutura Quadro.
|— src
|   |— main.c      Contém a função principal e a lógica de interação com o usuário.
|   |— memoria.c  Função de visualização da memória física.
|   |— processo.c Funções para criação de processos e visualização de tabelas de páginas.
|   |— quadro.c   Função de criação dos quadros de memória e gestão do vetor de quadros livres.
```

A seguir apresento uma visão generalista do que foi implementado dentro de cada um dos arquivos.

main.c

Arquivo principal onde a execução do programa começa.

Principais funções e variáveis

Processo *processos[10];: Um array para armazenar até 10 processos.
int num_processos = 0;: Contador para o número de processos criados.

main()

- Função principal do programa.
- Inicializa a semente para geração de números aleatórios com `srand(time(NULL));`.
- Chama a função `alocar_quadros` para inicializar os quadros livres.
- Loop principal que apresenta um menu de opções para o usuário
- Visualizar memória física (`visualizar_memoria_fisica`).
- Criar um novo processo (`cria_processo`).

- Visualizar a tabela de páginas de um processo específico (visualizar_tabela_paginas).
- Sair do programa.

memoria.c

Responsável pelas funções de gerenciamento e visualização da memória física.

Principais funções e variáveis:

- int uso_quadro[TAMANHO_MEMORIA_FISICA / TAMANHO_PAGINA] = {0};: Array que rastreia o uso de cada quadro.
- visualizar_memoria_fisica(): Função que exibe o estado atual da memória física:
- Calcula a quantidade de memória usada e livre.
- Exibe detalhes sobre cada quadro (ocupado ou livre e a quantidade de memória usada).

memoria.h

Contém definições e declarações relacionadas à memória.

Principais definições

- TAMANHO MEMORIA FISICA: 65536 bytes.
- TAMANHO_MEMORIA_LOGICA: 32768 bytes.
- TAMANHO_PAGINA: 4096 bytes.

Variáveis e funções globais

- extern int uso_quadro[];
- void alocar_quadros(int tamanho_memoria_fisica, int tamanho_pagina);
- void visualizar_memoria_fisica();

processo.c

Criar e gerenciar processos.

Principais funções

cria_processo()

Função para criar um novo processo

- Gera um tamanho aleatório para o processo dentro do tamanho da memória lógica.
- Aloca memória para a estrutura Processo.
- Calcula o número de páginas necessárias.
- Verifica se há quadros disponíveis suficientes.
- Inicializa a tabela de páginas do processo e aloca memória para cada página.
- Associa cada página a um quadro aleatório disponível.

- Atualiza o uso do quadro e remove-o da lista de quadros livres se estiver totalmente ocupado.

visualizar_tabela_paginas()

Função para exibir a tabela de páginas de um processo

- Exibe detalhes do processo e o mapeamento entre suas páginas e os quadros físicos.

processo.h

Definições e declarações relacionadas a processos.

Estruturas

Processo: Representa um processo, contendo ID, tamanho, número de páginas e a tabela de páginas.

Funções globais

- Processo* cria_processo(int id_processo, int tamanho_memoria_fisica, int tamanho_memoria_logica, int tamanho_pagina);
- void visualizar_tabela_paginas(Processo *process);

pagina.h

Este arquivo contém definições relacionadas a páginas.

Principais definições

- Estrutura Pagina: Representa uma página, contendo ID, ponteiro para a memória, número do quadro e memória usada.

quadro.c

Gerencia os quadros de memória.

Principais funções e variáveis

- Quadro *quadros_livres = NULL;; Lista encadeada de quadros livres.
- aloca_quadros(): Função para inicializar a lista de quadros livres:
- Aloca e inicializa quadros, adicionando-os à lista encadeada de quadros livres.
- remover_quadro_livre(): Função para remover um quadro da lista de quadros livres:
- Remove um quadro específico da lista encadeada, ajustando os ponteiros para manter a lista.

quadro.h

Definições e declarações relacionadas a quadros de memória.

Principais definições

- Estrutura Quadro: Representa um quadro de memória, contendo número, espaço usado e ponteiro para o próximo quadro na lista.
- Variáveis e funções globais:
- extern Quadro *quadros_livres;
- void alocar_quadros(int tamanho_memoria_fisica, int tamanho_pagina);
- void remover_quadro_livre(int numero_quadro);

Para execução do programa

Organizar os arquivos da mesma maneira em que previamente foi apresentado e ter todos os arquivos cabeçalho usados no programa (stdio.h, stdlib.h, stdbool.h & time.h).

Doravante, enquanto na raiz do projeto, executar no terminal o comando “make” para iniciar a compilação do programa. Após a compilação bem sucedida, execute o programa com “./main”.

Caso queira limpar os arquivos objetos e o executável gerados pelo Makefile, execute no terminal “make clean”.

Casos de Teste

- Criar e Visualizar um Processo
 - Escolha a opção 2. Criar processo.
 - Insira o ID do processo (por exemplo, 1).
 - O programa criará um processo com um tamanho aleatório e alocará as páginas de memória necessárias.
 - Escolha a opção 3. Visualizar tabela de páginas de um processo.
 - Insira o ID do processo criado (por exemplo, 1).
 - O programa mostrará a tabela de páginas do processo, indicando quais páginas estão mapeadas para quais quadros.

Saída Esperada

Informe o número do processo: 1

Processo 1 criado com tamanho de X bytes.

Informe o número do processo: 1

== Processo 1 ==

Tamanho do processo: X bytes

Número de páginas: Y

Mapeamento entre páginas e quadros:

Página 0 -> Quadro A (Memória usada: B bytes)

Página 1 -> Quadro C (Memória usada: D bytes)

- Visualizar Memória Física
 - Escolha a opção 1. Visualizar memória.
 - O programa mostrará a quantidade de memória usada e livre, bem como o estado de cada quadro.

Saída Esperada

===== *Memória física* =====

Memória usada | *N bytes (M%)*

Memória livre | *P bytes (Q%)*

===== *Quadros* =====

Quadro 0 | *Ocupado* | *A/B bytes*

Quadro 1 | *Livre* | *C/D bytes*