

Trabalho Prático 1

CCF221 (AEDS 1)

Sistema de gerenciamento de processos utilizando lista de cursores

INTEGRANTES: 01- Arthur Fernandes Bastos. [EF04679]

02- Ângelo Cupertino Machado. [EF04695]

03- Gabriel Ryan dos Santos. [EF04688]

Sumário

1- Introdução e Objetivos.	- 03
2- Preparação para o Desenvolvimento.	- 04
3- Desenvolvimento do Código.	- 05
- 3.1 Parte 1	- 05
- 3.2 Parte 2	- 06
4- Resultados	- 07
5- Passos finais.	- 07
6- Concluindo.	- 08

Introdução e Objetivos

O Trabalho Prático 1 da disciplina AEDS 1 tem como objetivo principal colocar em prática os conhecimentos adquiridos sobre Tipo Abstratos de Dados (TAD), alocação dinâmica, vetores, ponteiros, listas duplamente encadeadas e além disso possibilitar os alunos a aprender e entender o uso de Cursores para a ordenação dentro de um vetor.

O trabalho vem com a intenção também de demonstrar na prática as várias possibilidades que a Linguagem C oferece. Além de incentivar o desenvolvimento em equipe que irá ser crucial para o futuro dos alunos no mercado de trabalho. O principal objetivo dessa atividade é criar um TAD-Processos utilizando Cursores que tem como finalidade a inserção de valores dentro de células alocadas dinamicamente na memória e a ordenação das mesmas dentro dele. Dentro deste ponto, os alunos têm de realizar os seguintes itens:

- **a)** Criar uma área de memória interna vazia. Aloca um vetor de células de tamanho N. (Utilizando a alocação dinâmica deste vetor, devido aos altos valores de N que serão adotados);
- **b)** Obter o número de células ocupadas na área de memória. Retorna o número de posições ocupadas no vetor;
- **c)** Inserir um item de dado na área interna de memória, mantendo os itens ordenados;
- **d)** Retirar o primeiro item da área de memória;
- **e)** Imprimir o conteúdo da área de memória. (Somente imprime o conteúdo das células ocupadas).

Ademais,

- Não se deve fazer a ordenação através dos ponteiros, limitando-se ao uso dos Cursores.
- Utilizar uma função para demonstrar o tempo que leva para o processamento dos casos teste, visando não ultrapassar o tempo limite.

Preparação para o Desenvolvimento

- 1** - O trabalho foi desenvolvido e versionado por meio da plataforma GitHub, Sendo assim, o primeiro passo foi criar um projeto e compartilhá-lo entre o grupo.
- 2** - Em seguida foi adicionado os objetivos do trabalho e foi discutido no grupo como seria feito a padronização dos commits e como nós iríamos começar a desenvolver o projeto.
- 3** - Foi adicionado os casos testes e um resumo de o que seria feito.
- 4** - Foi feito através do README.md uma lista de progressos com o objetivo de marcar o andamento do projeto e demonstrar o que ainda faltava para ser desenvolvido.
- 5** - Estudo pela monitoria para entender o conceito de Cursores e como eles deveriam ser utilizados no sistema.
- 6** - Início do Código e desenvolvimento do sistema.

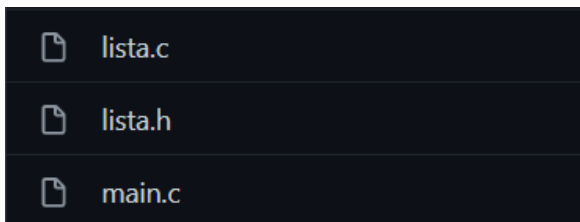
Desenvolvimento do Código Parte 1

1 - A primeira parte a ser feita foi o TAD - Processos:

Esse TAD tem o objetivo de gerenciar a criação de um processo, no qual será implementado o TAD lista, juntamente das funcionalidades e dará início ao procedimento a ser realizado. O próximo passo foi adequar as funções de setar o tempo na hora de iniciar, o PID (Número aleatório) e a Prioridade no processo:

```
void setPid(Processo *processo);  
  
int getPid(Processo *processo);  
  
void setPrioridade(Processo *processo);  
  
int getPrioridade(Processo *processo);  
  
void setHoras(Processo *processo);  
  
char* getHoras(Processo *processo);
```

2 - Em seguida foi criado o TAD-Lista e um protótipo da função main:



A screenshot of a file explorer window with a dark background. It displays three files, each with a document icon to its left: 'lista.c', 'lista.h', and 'main.c'.

No processo de desenvolvimento dessa parte o grupo conseguiu realizar os seguintes objetivos necessários:

- ☒ Criar um tad para processos
- ☒ Criar uma área de memória interna vazia. Aloca um vetor de células de tamanho N, vamos utilizar alocação dinâmica deste vetor, devido aos altos valores de N que adotaremos.
- ☒ Obter o número de células ocupadas na área de memória. Retorna o número de posições ocupadas no vetor.
- ☐ Inserir um item de dado na área interna de memória, mantendo os itens ordenados.
- ☐ Retirar o primeiro item da área de memória.
- ☒ Imprimir o conteúdo da área de memória. Somente imprime o conteúdo das células ocupadas.

Desenvolvimento do Código Parte 2

Após ter completado os itens anteriores o próximo passo foi construir uma função que inserisse um novo valor dentro de uma célula vazia e ordenasse o vetor após esse ato. Como era um processo de lógica complicado, os integrantes do grupo se reuniram apenas para destrinchar o que deveria ser feito para que o sistema compreendesse o que deveria ser feito. Depois de muitas discussões para a melhor solução, foi alcançado o seguinte protótipo do que deveria ser feito pela equipe:

```
Inserir o valor $a
1- $a é o maior
  //sim
    entao ele eh o maior valor, e o anterior do $a recebe o indice do $ultimo e
    o proximo do $ultimo recebe o indece do $a

    proximo de $a = -1
    $ultimo = posicao de $a
  //nao
    $a é o menor?
    //sim
      entao ele eh o menor valor, e o proximo do $a recebe o indice do $primeiro e
      o anterior do $primeiro recebe o indece do $a

      anterior de $a = -1
      $primeiro = posicao de $a
    //nao
      2 - $b == valor a ser comparado
      pega o valor do $b.
      tem algum valor entre o $a e o $b?
      //sim
        pega esse valor $c e volta para o item 2.
      //nao
        anterior de $b recebe a posicao $a e o proximo de $a recebe
        a posicao de $b

        $c = v[$b->proximo] // aponta pra c

        proximo de $b recebe posicao de $a,
        anterior de $a recebe a posicao de $b,
        proximo de $a recebe $b->proximo, // ou seja prox $a recebe $c

        v[$c]->anterior = posicao de $a,
```

E após implementar essa lógica como um código foi possível inserir um valor e realizar a ordenação das células dentro do vetor, possibilitando assim seguir para o próximo passo que seria a remoção de algum valor pedido, mantendo a ordenação. Após a criação da função de remoção e adequação das duas dentro da main o projeto estava completando todos os requisitos e pudemos assim seguir para a fase de testes.

Resultados

Com todos os objetivos concluídos, o próximo passo foi implementar os testes no sistema criado para ver se estava tudo funcionando corretamente. Após vários testes manuais e correções no código passamos para dar entrada com os casos teste disponibilizados pela professora.

O primeiro resultado foi:

Teste 01 : (100.000 Inserções e 100.000 retiradas)

```
1 Nome/Numero do teste: teste100000 Tempo de execucao: 53.158000
```

Teste 02 : (200.000 Inserções e 200.000 retiradas)

```
1 Nome/Numero do teste: teste200000 Tempo de execucao: 232.131000
```

Teste 03 : (300.000 Inserções e 300.000 retiradas)

```
1 Nome/Numero do teste: teste300000 Tempo de execucao: 562.914000
```

Teste 04 : (400.000 Inserções e 400.000 retiradas)

```
1 Nome/Numero do teste: teste400000 Tempo de execucao: 904.149000
```

Teste 05 : (500.000 Inserções e 500.000 retiradas)

```
1 Nome/Numero do teste: teste500000 Tempo de execucao: 1478.515000
```

Teste 06 : (600.000 Inserções e 600.000 retiradas)

```
1 Nome/Numero do teste: teste600000 Tempo de execucao: 2208.422000
```

Concluindo

Após ter feito os testes e comparar com o tempo de outras pessoas da turma, foi feito um “Pente Fino” no código, deixando assim comentários para facilitar a leitura e corrigir possíveis detalhes que ainda estavam despercebidos e também passar o olho em algumas medidas que podem deixar o código um pouco mais leve.

O sistema está disponível para acesso no Github:

(<https://github.com/acmachado14/trabalho-aeds1>)

Para o teste em sua máquina deve-se baixar a pasta com os arquivos e para realizar algum dos Testes deixados nos Resultados:

- **Deve executar os seguintes comandos:**
 - `cd src`
 - `gcc processo.c main.c lista.c`
 - `./a.exe`
- **A entrada deve ser : “ testeX00000 ”**
*Onde X é o número do teste que deseja executar.

Caso queira dar entrada em um teste que não está dentro dos apresentados, crie um novo arquivo de teste na pasta: testes, e logo após digite os mesmos comandos :

- **Crie um teste na pasta de testes com nome: nomedoteste.txt**
- **Deve executar os seguintes comandos:**
 - `cd src`
 - `gcc processo.c main.c lista.c`
 - `./a.exe`
- **A entrada deve ser (nomedoteste)**
- **Aguarde o procedimento.**

Por fim, podemos concluir que com esse trabalho foi possível observar de perto como é feita a alocação de memória e como pode ser utilizada, além disso, foram demonstrados dois novos conceitos de ordenação de itens em um vetor que é o uso dos Cursores e das listas duplamente encadeadas.