Universidade Estadual do Norte do Paraná - UENP
1º Trabalho Computacional de Estruturas de Dados

1 Objetivo

Este trabalho tem como objetivo praticar o uso de tipos abstratos de dados e estruturas do tipo Lista.

2 Regras Importantes

- Não é tolerado plágio. Trabalhos copiados serão penalizados com zero.
- A data de entrega é inadiável. Para cada dia de atraso, é retirado um ponto da nota do trabalho.

3 Relatório

O relatório do trabalho, que deve conter:

- Introdução: descrição do problema a ser resolvido e visão geral sobre o funcionamento do programa (em termos de módulos, arquivos, etc.).
- Implementação: Devem ser descritas as estruturas de dados utilizadas (de preferência com diagramas ilustrativos), o funcionamento das principais funções utilizadas incluindo pré e pós condições, o formato de entrada e saída de dados, bem como decisões tomadas relativas aos casos e detalhes de especificação que porventura estejam omissos no enunciado. Modularize o seu programa como discutido em sala de aula.
- Conclusão: comentários gerais sobre o trabalho e as principais dificuldades encontradas em sua implementação.
- Bibliografia: bibliografia utilizada para o desenvolvimento do trabalho, incluindo sites da Internet se for o caso.

4 Envio

O trabalho deve ser enviado para o e-mail alan.floriano@uenp.edu.br até o dia 12/08/2023:

- O assunto da mensagem deve ser ed202301CC:trab1:nome1:nome2 Por exemplo: ed202301CC:trab1:joaosilva:mariacosta
- Documentação do trabalho (em formato PDF).
- Todos os arquivos .c e .h criados (exigido código muito bem documentado!)
- O makefile.
- Favor nomear os arquivos da seguinte maneira: estoque.c, estoque.h, trabalho1.c,

5 Trabalho

Você é o responsável pelo estoque produtos de uma grande rede de supermercados. A partir de um arquivo de entrada (”entrada.txt”) seu sistema deve ser capaz de realizar o processamento do estoque do supermercado que você está gerenciando.

Nesse trabalho, você deverá implementar a estrutura produto, que irá conter o nome (char[20]), código de identificação (int), valor(float) e data de validade (int[3]). E para armazernar esses produtos é necessario ter a estrutura lista, que será alocada de forma dinâmica.

```
    struct produto{
        char nome[20];
        int codigo;
        float valor;
        int data[3];
    };

    struct lista{
        Produto* p;
        Lista* prox;
    };
```

Neste trabalho você deve implementar as estruturas e operações do TAD estoque.h, que estão descritas a seguir:

```
//Tipo que define o produto
typedef struct produto Produto;
//Tipo que define a lista
typedef struct lista Lista;
/*Cria um novo produto*/
Produto* CriaProduto(char* nome,int codigo,float valor,int*
    data_de_validade);
Lista* CriaLista();
/*Insere um produto em uma lista*/
Lista* InsereListaProduto (Lista* l, Produto* p);
```

```
/*Retira um produto de uma determinada lista*/
Lista* RetiraListaProduto (Lista* l, int id_produto);
/*Verifica se um produto est  presente em uma determinada lista*/
int VerificaListaProduto (Lista* p, int id_produto);
/*Verifica se existe um produto est  vencido em uma determinada
   lista*/
Lista* VerificaListaValidade (Lista* p);
/*Imprime todos os produtos de uma lista*/
void ImprimeListaProdutos (Lista* p);
/*Orderna Lista pelo valor do produto*/
Lista* OrdenaListaValor (Lista* p);
/*Orderna Lista pelo valor do produto*/
Lista* OrdenaListaVencimento (Lista* p);
```

5.1 O Programa Testador (trabalho1.c)

O programa testador deverá ser capaz de ler as instruções do arquivo texto de entrada e realizar as devidas operações no tad estoque. O seu programa (trabalho1.c) deverá ler os dados de entrada a partir de um arquivo de entrada (”entrada.txt”). O arquivo de entrada é basicamente uma lista de comandos (um por linha) em formato texto. O último comando é a palavra FIM, que indica o final do arquivo. O formato a ser usado é exemplificado abaixo:

```
PRODUTO ARROZ 123 18.3 12 12 2023
PRODUTO FEIJAO 234 8.3 13 12 2023
PRODUTO PAO_DE_FORMA 114 7.0 23 06 2023
PRODUTO SABAO 132 1.2 23 12 2025
RETIRA 123
IMPRIME_LISTA
ATUALIZA_PRECO 234 9.4
VERIFICA_VALIDADE 13 07 2023
VERIFICA_LISTA 123
ORDENA_LISTA_VALIDADE
IMPRIME_LISTA
VERIFICA_VALIDADE 20 06 2023
ORDENA_LISTA_VALOR
IMPRIME_LISTA
FIM
```

E como saída é esperado um arquivo de texto (”saida.txt”), com as seguintes mensagens:

```
PRODUTO ARROZ 123 ADICIONADO
PRODUTO FEIJAO 234 ADICIONADO
PRODUTO PAO_DE_FORMA 114 ADICIONADO
PRODUTO SABAO 132 ADICIONADO
PRODUTO ARROZ 123 RETIRADO
SABAO 132 1.2 23 12 2025
PAO_DE_FORMA 114 7.0 23 06 2023
```

```
FEIJAO 234 8.3 13 12 2023
PRECO ATUALIZADO FEIJAO 234 9.4
PRODUTO PAO_DE_FORMA 114 VENCIDO
PRODUTO NAO ENCONTRADO NA LISTA
PAO_DE_FORMA 114 7.0 23 06 2023
FEIJAO 234 8.3 13 12 2023
SABAO 132 1.2 23 12 2025
PRODUTO VENCIDO NAO ENCONTRADO NA LISTA
SABAO 132 1.2 23 12 2025
PAO_DE_FORMA 114 7.0 23 06 2023
FEIJAO 234 8.3 13 12 2023
```

OBSERVAÇÃO:

Ao final, o programa deve esvaziar a memória, eliminando assim todos os itens alocados.