(2021-1) P1-Ex02

Desenvolver um TAD para representar um produto. O TAD deve gerenciar as seguintes informações do produto:

- código: identificador numérico do produto
- descrição: descrição do produto
- últimos preços: últimos 4 preços praticados.

As funcionalidades a ser implementadas são:

- Criar um produto
- Destruir um produto
- Acessar o código do produto
- Acessar a descrição do produto
- Acessar o preço atual do produto
- Calcular a média dos últimos preços praticados (no máximo 4)
- Alterar o preço do produto

TAREFA

Desenvolver as funções contidas no arquivo tad_produto.c. Submeta essa arquivo no run.codes

Arquivos

tad_produto.h

```
#ifndef _TAD_PRODUTO_
#define _TAD_PRODUTO_

#include <stdbool.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

/*****************************************
 * DADOS
*****************************************/
typedef struct produto Produto;
```

```c
/*****************************************
 * OPERAÇÕES
*****************************************/

/**
 * Cria dinamicamente o produto e preenche os atributos
 * Parâmetro codigo: Código do produto a ser criado
 * Parâmetro descricao: Descrição do produto a ser criado
 * Parâmetro preco: Preço do produto a ser criado
 * RETORNO: endereço do Produto criado
 */
Produto* produto_criar(int codigo, char* descricao, double preco);

/**
 * Desaloca o Produto criado
 * Parâmetro endProduto: Endereço da variável que contém o ponteiro para o Produto a ser desalocado.
 * A função também deve limpar o lixo de memória no endereço da variável recebido por parâmetro
 */
void produto_destruir(Produto** endProduto);

/**
 * Como os atributos do Produto estão encapsulados, essa função tem como objetivo prover o acesso
 * ao código do produto p. Portanto, a função devolve o código do produto p.
 * Parâmetro p: Ponteiro para a struct Produto.
 * RETORNO: o código do Produto p ou -1 em caso de erro.
 */
int produto_codigo(Produto* p);
```

```c
/**
 * Como os atributos do Produto estão encapsulados, essa função tem como objetivo prover o acesso
 * a descrição do Produto p. Portanto, a função devolve a descrição do produto p.
 * A função deve usar a estratégia do scanf, copiando a string no endereço "saida" recebido por parâmetro.
 * Parâmetro p: Ponteiro para a struct Produto.
 * Parâmetro saida: Endereço de memória em a função deve copiar a string correspondente a descrição do produto.
 * RETORNO: true se a cópia foi realizada com sucesso ou false caso contrário.
 */
bool produto_descricao(Produto* p, char* saida);

/**
 * Como os atributos do Produto estão encapsulados, essa função tem como objetivo prover o acesso
 * ao preço do produto p. Portanto, a função devolve o preço atual do produto p.
 * Parâmetro v: Ponteiro para a struct Produto.
 * RETORNO: o código do Produto p ou -1 em caso de erro.
 */
double produto_preco(Produto* p);

/**
 * Calcula a média dos últimos preços praticados (vetor ultimos_precos). Lembrando que o produto pode ter,
 * no máximo, os últimos 4 preços do produto.
 * Parâmetro v: Ponteiro para a struct Produto.
 * RETORNO: a média dos preços ou -1 em caso de erro.
 */
double produto_media_preco(Produto* p);
```

```c
/**
 * Altera o preço do produto. A alteração implica na manipulação do vetor ultimos_precos. Esse vetor pode
 * armazenar no máximo 4 valores. A função deve garantir que nesse vetor seja mantido sempre os últimos preços atribuídos ao produto.
 * Portanto, faça o gerenciamento correto desse vetor.
 * Parâmetro v: Ponteiro para a struct Produto.
 * Parâmetro preco: Preço a ser atribuído ao produto.
 * RETORNO: true caso a atribuição ocorra com sucesso e false caso contrário.
 */
bool produto_altera_preco(Produto* p, double preco);

#endif
```

tad_produto.c

```c
#include "tad_produto.h"

struct produto{
    int codigo;
    char descricao[50];
    double ultimos_precos[4]; // armazena no máximo os últimos 4 preços praticados
    int qtd_precos;           // quantidade de preços armazenados no vetor ultimos_precos
};


Produto* produto_criar(int codigo, char* descricao, double preco);
void produto_destruir(Produto** endProduto);
int produto_codigo(Produto* p);
bool produto_descricao(Produto* p, char* saida);
double produto_preco(Produto* p);
double produto_media_preco(Produto* p);
bool produto_altera_preco(Produto* p, double preco);
```

main.c

Faça os seus testes aqui.

```c
#include "tad_produto.h"


int main(){


    return 0;
}
```