IGCE – Instituto de Geociências e Ciências Exatas
DEMAC – Departamento de Estatística, Matemática e Computação

1° Trabalho da disciplina de Estrutura de Dados I (noturno)

Prof. Dr. Glauco Vitor Pedrosa

1. Usando uma estrutura de **Lista <u>Estática</u> ORDENADA**, implemente as funções solicitadas abaixo usando a linguagem C e considere que o campo *dados* da nossa estrutura para Lista seja o seguinte:

```
#define Max 50

struct produto{
       int codProd;         //código do produto
       char nomeProd[10];   //nome do produto
       float valor;         //valor do produto
       int qtdeEstoque;     //quantidade disponível em estoque
}

typedef struct produto Produto;

struct Lista{
       Produto dados[Max]; //arranjo com os elementos da lista
       int FL;             // índice da primeira posição livre da lista
};

typedef struct Lista Lista;
```

   a) Implemente uma função para inserir produtos *<u>ordenados pelo código</u>* em uma lista. Essa função recebe como parâmetro um tipo de dado Produto e retorna 1 se o produto foi inserido com sucesso na lista ou 0 se houve algum problema na inserção do produto na lista, por exemplo, a lista está cheia, lista não inicializada e/ou código do produto já está cadastrado. O protótipo dessa função é definido por:

```
int inserir_produto (Lista* L, Produto p);
```

   b) Função para remover os n primeiros elementos da lista. A função recebe como entrada o endereço da lista, e o valor de n (um número inteiro não negativo). Caso n seja maior que o tamanho da lista, todos os elementos da lista são removidos e a lista resultante deve ser vazia, Caso a operação seja executada com sucesso retorne (1), caso contrário, retorne (0). O protótipo da função é definido por:

```
int remover_produto (Lista *L, int n);
```

   c) Função para trocar dois elementos de posição. A função recebe como entrada o endereço da lista, e o índice de duas posições da lista. Caso a lista exista, possua pelo menos dois elementos, e as posições sejam válidas, **<u>o campo nomeProd</u>** das posições indicadas são trocados entre si e a função retorna sucesso (1). Caso contrário, a função retorna insucesso (0). O protótipo da função é definido por:

```
int trocaProdutos (Lista *L, int pos1, int pos2);
```

   d) Implemente uma função para buscar a posição e as informações do produto com menor numero de peças em estoque. A função recebe como entrada o endereço da lista e retorna a posição e o produto que tem menor quantidade de peças em estoque. Caso a lista não exista ou não possua elementos, a função retorna insucesso (0). Caso seja possível encontrar o maior elemento a função retorna sucesso. O protótipo da função é definido por:

```
int busca_produto (Lista *L, int  *pos, Produto *MenorEstoque);
```

e) Implemente uma função para efetuar a compra de um produto. A função recebe como parâmetro o código do produto a ser comprado e a quantidade a ser comprada. Caso o produto exista e a quantidade em estoque seja suficiente para finalizar a compra, a função retorna um inteiro indicando sucesso (1) e decrementa a quantidade em estoque desse produto de acordo com o que foi comprado. Caso haja alguma falha, lista inexistente, lista vazia, produto inexistente ou quantidade insuficiente em estoque, a função retorna insucesso (0). O protótipo é definido por:

```
int efetuarCompra(Lista *L, int codProd, int qtde);
```

f) Implemente uma função para imprimir a lista de produtos:

```
void imprimeLista(Lista *L);
```

2. Implementar um programa para manipulação de polinômios do tipo

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x^1 + a_0$$

Para tal, o polinômio deve ser armazenado usando uma **Lista Dinâmica Encadeada Ordenada**, sendo que cada elemento da lista deve armazenar o n-ésimo termo do polinômio (diferente de 0), e deve conter o valor da potência de (inteiro) e o coeficiente correspondente (inteiro). Por exemplo, o polinômio:

$$P(x) = 3x^6 - 2x^4 + x^3 + 5x + 2$$

deve ser representado pela lista:

Fica a critério do aluno a escolha da técnica de implementação, tanto em relação à forma de alocação, quanto à forma de agrupamento. A utilização de nó descritor, encadeamento cíclico/simples/duplo também fica a critério do aluno. Deve ser criada uma interface que permita ao usuário executar qualquer uma das operações a seguir, a qualquer momento:

a) Inicializar um polinômio.

$$P(x) = 0x^0$$

b) Inserir um termo no polinômio existente.

O usuário deverá entrar com o coeficiente e o grau do polinômio. Por exemplo, se o usuário digitar: 5 2, significa adicionar o termo $5x^2$ ao polinômio existente. Além disso, se já existe um termo no polinômio, o valor do coeficiente do novo termo deve ser adicionado ao já existente, assim:

$$P(x) = a_n x^n + \ldots + (a_k + a_k')x^k + \cdots + a_1 x^1 + a_0$$

c) Imprimir *P(x)*

Se o polinômio for

$$P(x) = 2x^5 - 3x^2 + 7$$

a sua representação interna será:

A seguinte expressão deverá ser visualizada na tela: **+7 -3x^2 +2x^5**

d) Eliminar o termo associado à n-ésima potência.

Se o polinômio atual for

$$P(x) = 2x^5 - 3x^2 + 7$$

e o usuário solicitar a remoção do termo associado à potência 2, o polinômio resultante será *P(x) = $2x^5+7$*,e o nó referente à potência de deve ser liberado resultando na estrutura:

e) Reinicializar um polinômio.

Fazer *P(x)=$0x^0$* e liberar os nós do *P(x)* anterior.

f) Calcular o valor de P(x) para um dado valor de x.

Por exemplo, se o polinômio atual for

$$P(x) = 3x^6 - 2x^4 + x^3 + 5x + 2$$

e o usuário solicitar o cálculo de P(x) para x = 2, o valor de P(2) deve ser calculado: *P(2)= $2*2^6$ – $2*2^4$ + 2^3 + 5*2 + 2 = 180* e o resultado deve ser apresentado na tela.

3. Escreva um programa para verificar se uma expressão matemática tem os parênteses agrupados de forma correta, isto é:
Condição 1: se o número de parênteses abertos e fechados são iguais e;
Condição 2: se todo parêntese fechado foi precedido por um parêntese aberto.
Por exemplo:
As expressões **((A+B)** ou **(A+B(** violam a condição 1, pois a quantidade de parênteses aberto é diferente da quantidade de parênteses fechados;
As expressões **)A+B(– C** ou **(A+B) –)C + D(** violam a condição 2, pois existe a ocorrência de um parêntese fechado sem ter havia a ocorrência de um parêntese aberto. Note que nesses exemplos, eles satisfazem a condição 1.

Verificar se alguma das 2 condições é violada, se sim, indicar qual condição é violada

Implemente a seguinte TAD Pilha na construção desse programa:

```
#define Max 50

struct pilha {
     char pilha[MAX];
     int topo;
};

typedef struct pilha Pilha;

int pilha_cheia(Pilha p); // retorna 1 se pilha cheia, 0 caso contrário
int pilha_vazia(Pilha p);  // retorna 1 se pilha vazia, 0 caso contrário
void cria_pilha(Pilha *p); // inicializa uma pilha
void push(Pilha *p, char elem); // empilha um elemento
char pop(Pilha *Ptp); // desempilha um elemento
void libera_pilha(Pilha *Ptp); // esvazia a pilha
```

O programa deverá conter todas as operações acima

4. Considerando uma estrutura de dados **Fila com alocação dinâmica**, escreva um programa que, dada uma fila F com números inteiros (fornecidos pelo usuário), este programa deve separar todos os valores guardados em F de tal forma que os valores múltiplos de 3 são movidos para uma nova fila chamada F3 e os valores múltiplos de 2 são movidos para uma fila chamada F2 e os demais valores são movidos para uma fila F4. Lembre-se que um numero pode ser múltiplo de 2 e 3 simultaneamente. Nesse caso esse valor dever ser inserido nas duas listas, F2 e F3. Exiba a fila original F e no final do processamento, exiba F, F2, F3 e F4.