INSTITUTO FEDERAL
Minas Gerais
Campus São João Evangelista
Bacharelado em Sistemas de Informação

Atividade 8- Lista Dinâmica

Pedro Henrique Santos

Uma lista com alocação dinâmica permite armazenar os dados de forma flexível. Em vez de reservar um espaço fixo, a memória é alocada conforme necessário. O item tem um ponteiro que aponta para o próximo, isso permite adicionar, remover ou ver os itens de forma eficiente, assim ela gerencia a memória durante a execução do programa, evitando desperdício.

Código de lista de alocação dinâmica comentado:

```
#include <iostream>
#include <string>

using namespace std;

// Struct do item da lista
struct Item
{
    int idade;
    string nome;
    Item *proximo; // Ponteiro para o próximo item na lista
};

// Struct da lista
struct Lista
{
    Item *primeiro = NULL; // Ponteiro para o primeiro item da lista
    Item *ultimo = NULL;   // Ponteiro para o último item da lista
};
Lista *L = NULL;

// Função para criar a lista
void criarLista()
{
    L->primeiro = new Item; // Criação do primeiro item da lista
    L->ultimo = L->primeiro;
    L->primeiro->proximo = NULL; // Inicialização do próximo como nulo,
pois é o único item na lista inicialmente
}

// Função para verificar se a lista está vazia
int vazia()
{
    return (L->primeiro == L->ultimo); // A lista está vazia se o
primeiro item for igual ao último
```

```
}

// Função para inserir um item no final da lista
void inserirUltima()
{
    Item *x = new Item;

    cout << "Digite um nome: ";
    cin >> x->nome;
    cout << "Digite a idade: ";
    cin >> x->idade;

    L->ultimo->proximo = x;    // O último item passa a apontar para o
novo item
    L->ultimo = x;             // O novo item torna-se o último
    L->ultimo->proximo = NULL; // Garante que o próximo do último seja
nulo, indicando o fim da lista
}

// Função para mostrar todos os itens da lista
void mostrar()
{
    Item *aux;
    aux = L->primeiro->proximo; // Começa do primeiro item da lista
    while (aux != NULL)
    {
        cout << "Nome: " << aux->nome << " "
             << "Idade: " << aux->idade << endl;
        aux = aux->proximo; // Move para o próximo item na lista
    }
}

// Função para inserir um item no início da lista
void inserirPrimeira()
{
    Item *x = new Item;

    cout << "Digite um nome: ";
    cin >> x->nome;
    cout << "Digite a idade: ";
    cin >> x->idade;

    if (!vazia())
    {
        x->proximo = L->primeiro->proximo; // O novo item aponta para o
que era o primeiro item
        L->primeiro->proximo = x;          // O primeiro item agora é o
novo item
    }
```

```
    else
    {
        L->ultimo->proximo = x; // Se a lista estava vazia, o último item
também deve apontar para o novo item
        L->ultimo = x;          // O novo item torna-se o último
        L->ultimo->proximo = NULL;
    }
}

// Função para inserir um item em uma posição específica da lista
void inserirPosicao(int n)
{
    int i = 0;
    Item *aux = L->primeiro;
    while (i < (n - 1) && aux != NULL)
    {
        i++;
        aux = aux->proximo;
    }

    Item *x = new Item;
    cout << "Digite um nome: ";
    cin >> x->nome;
    cout << "Digite a idade: ";
    cin >> x->idade;

    x->proximo = aux->proximo; // O novo item aponta para o próximo item
na posição
    aux->proximo = x;          // O item na posição aponta para o novo
item
}

// Função para remover o primeiro item da lista
void removerPrimeira()
{
    if (!vazia())
    {
        Item *aux = L->primeiro->proximo;
        cout << endl
             << "+++ REMOVENDO +++" << endl;
        cout << aux->nome << " " << aux->idade << endl
             << endl;
        L->primeiro->proximo = aux->proximo; // O primeiro item passa a
apontar para o próximo item
        if (aux == L->ultimo)
        {
            L->ultimo = L->primeiro; // Se o item removido era o último,
o primeiro também é o último
        }
```

```
        delete aux; // Libera a memória do item removido
    }
    else
    {
        cout << endl
             << "Vazia" << endl;
    }
}

// Função para remover o último item da lista
void removerUltima()
{
    if (!vazia())
    {
        Item *aux = L->primeiro;
        while (aux->proximo != L->ultimo)
        {
            aux = aux->proximo;
        }
        cout << endl
             << "+++ REMOVENDO +++" << endl;
        cout << aux->proximo->nome << " " << aux->proximo->idade << endl
             << endl;
        aux->proximo = NULL; // O item antes do último passa a apontar
para nulo
        delete L->ultimo;    // Libera a memória do último item
        L->ultimo = aux;     // O item anterior ao último torna-se o novo
último
    }
    else
    {
        cout << endl
             << "Vazia" << endl;
    }
}

// Função para remover um item de uma posição específica da lista
void removerPosicao(int posicao)
{
    if (!vazia())
    {
        int i = 1;
        Item *aux1 = L->primeiro, *aux2 = L->primeiro->proximo;
        while (i < posicao && aux2 != NULL)
        {
            i++;
            aux1 = aux2;
            aux2 = aux2->proximo;
        }
```

```
        if (aux2 == NULL)
        {
            cout << "Nada para remover" << endl;
        }
        else
        {
            cout << endl
                 << "+++ REMOVENDO +++" << endl;
            cout << aux2->nome << " " << aux2->idade << endl
                 << endl;
            if (aux2 == L->ultimo)
            {
                L->ultimo = aux1; // Se o item removido era o último, o
anterior torna-se o novo último
            }
            aux1->proximo = aux2->proximo; // O item anterior ao removido
passa a apontar para o próximo item
            delete aux2;                   // Libera a memória do item
removido
        }
    }
}

// Função para exibir o menu de opções
int menu()
{
    int opcao;
    cout << "++++ Opcoes ++++ " << endl;
    cout << "1. Inserir no inicio" << endl;
    cout << "2. Inserir no final" << endl;
    cout << "3. Inserir em uma posicao" << endl;
    cout << "4. Mostrar" << endl;
    cout << "5. Remover primeira" << endl;
    cout << "6. Remover ultima" << endl;
    cout << "7. Remover de uma posicao" << endl;

    cout << "0. Sair" << endl;
    cout << "Digite: ";
    cin >> opcao;
    return opcao;
}

// Main utilizada para redirecionar a função escolhida
int main()
{
    int opcao, p;
    L = new Lista;
    criarLista();
    do
```

```
    {
        opcao = menu();
        switch (opcao)
        {
        case 1:
            inserirPrimeira();
            break;
        case 2:
            inserirUltima();
            break;
        case 3:
            cout << "Digite a posicao: ";
            cin >> p;
            inserirPosicao(p);
            break;
        case 4:
            mostrar();
            break;
        case 5:
            removerPrimeira();
            break;
        case 6:
            removerUltima();
            break;
        case 7:
            cout << "Digite a posicao: ";
            cin >> p;
            removerPosicao(p);
            break;
        case 0:
            cout << "Saindo..." << endl;
            break;
        default:
            cout << "Selecione uma opção válida!" << endl;
            break;
        }
    } while (opcao != 0);
    return 0;
}
```