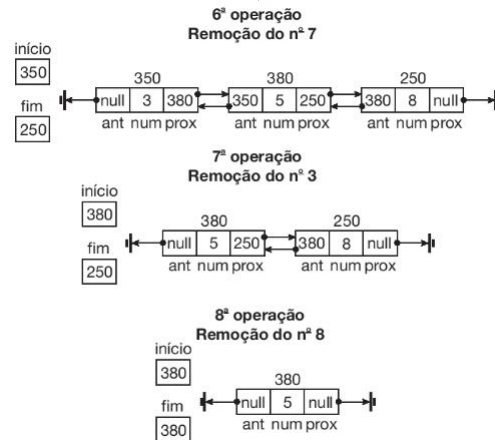
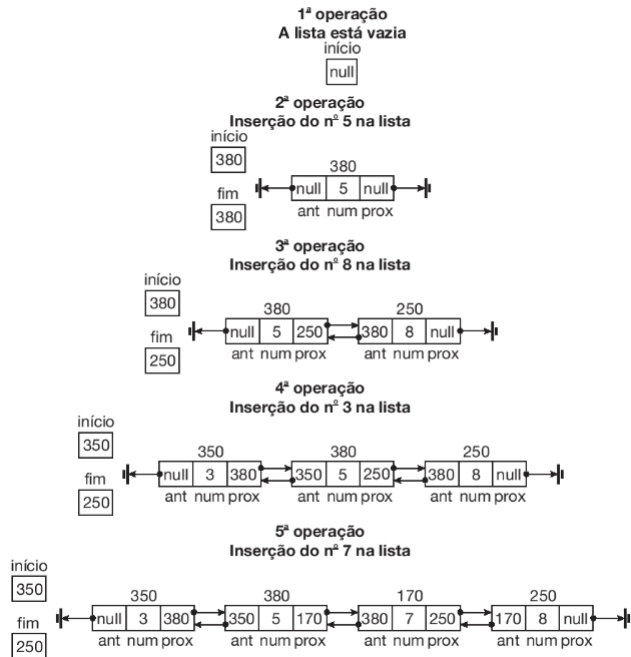


Lista duplamente encadeada ordenada

Murilo Dantas

Lista dupla enc. ordenada



Lista dup. ord.

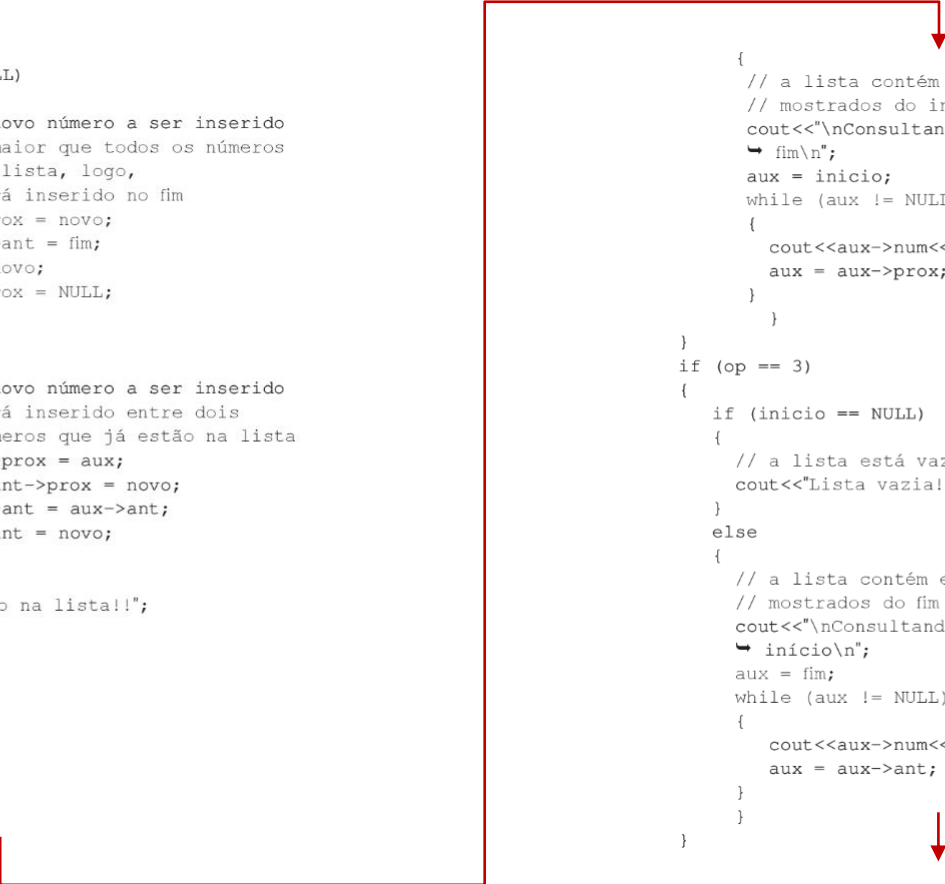
```
#include <iostream.h>
#include <conio.h>

void main()
{
    //Definindo o registro que representará
    //cada elemento da lista
    struct LISTA
    {
        int num;
        LISTA *prox;
        LISTA *ant;
    };
    // a lista está vazia, logo,
    // o ponteiro inicio tem o valor null
    // o ponteiro inicio conterá o endereço
    // do primeiro elemento da lista
    LISTA *inicio = NULL;
    // o ponteiro fim conterá o endereço
    // do último elemento da lista
    LISTA *fim = NULL;
    // o ponteiro aux é um ponteiro auxiliar
    LISTA *aux;
    // apresentando o menu de opções
    int op, numero, achou;
    do
    {
        clrscr();
        cout<<"\nMENU DE OPÇÕES\n";
        cout<<"\n1 - "Inserir na lista";
        cout<<"\n2 - Consultar a lista do início ao fim";
        cout<<"\n3 - Consultar a lista do fim ao início";
        cout<<"\n4 - Remover da lista";
        cout<<"\n5 - Esvaziar a lista";
        cout<<"\n6 - Sair";
        cout<<"\nDigite sua opção: ";
        cin>>op;
```

```
if (op < 1 || op > 6)
    cout<<"Opção inválida!!";
if (op == 1)
{
    cout<<"Digite o número a ser inserido na
    ↳ lista: ";
    LISTA *novo = new LISTA();
    cin>>novo->num;
    if (inicio == NULL)
    {
        // a lista estava vazia
        // e o elemento inserido será
        // o primeiro e o último
        novo->prox = NULL;
        novo->ant = NULL;
        inicio = novo;
        fim = novo;
    }
    else
    {
        // a lista já contém elementos
        // e o novo elemento
        // será inserido na lista
        // respeitando a ordenação crescente
        aux = inicio;
        while (aux != NULL && novo->num > aux->num)
        {
            aux = aux->prox;
        }
        if (aux == inicio)
        {
            // o novo número a ser inserido
            // é menor que todos os números da lista,
            // logo, será inserido no início
            novo->prox = inicio;
            novo->ant = NULL;
            inicio->ant = novo;
            inicio = novo;
        }
    }
}
```

Lista dupla enc. ordenada

```
else if (aux == NULL)
{
    // o novo número a ser inserido
    // é maior que todos os números
    // da lista, logo,
    // será inserido no fim
    fim->prox = novo;
    novo->ant = fim;
    fim = novo;
    fim->prox = NULL;
}
else
{
    // o novo número a ser inserido
    // será inserido entre dois
    // números que já estão na lista
    novo->prox = aux;
    aux->ant->prox = novo;
    novo->ant = aux->ant;
    aux->ant = novo;
}
}
cout<<"Número inserido na lista!!";
}
if (op == 2)
{
    if (inicio == NULL)
    {
        // a lista está vazia
        cout<<"Lista vazia!!";
    }
    else
```



```
{
    // a lista contém elementos e estes serão
    // mostrados do início ao fim
    cout<<"\nConsultando a lista do início ao
    fim\n";
    aux = inicio;
    while (aux != NULL)
    {
        cout<<aux->num<<" ";
        aux = aux->prox;
    }
}
if (op == 3)
{
    if (inicio == NULL)
    {
        // a lista está vazia
        cout<<"Lista vazia!!";
    }
    else
    {
        // a lista contém elementos e estes serão
        // mostrados do fim ao início
        cout<<"\nConsultando a lista do fim ao
        início\n";
        aux = fim;
        while (aux != NULL)
        {
            cout<<aux->num<<" ";
            aux = aux->ant;
        }
    }
}
```

Lista dupla enc. ordenada

```
if (op == 4)
{
    if (inicio == NULL)
    {
        // a lista está vazia
        cout<<"Lista vazia!!";
    }
    else
    {
        // a lista contém elementos
        // e o elemento a ser
        // removido deve ser digitado
        cout<<"\nDigite o elemento a ser
        ↳ removido: ";
        cin>>numero;
        // todas as ocorrências da lista,
        // iguais ao número digitado,
        // serão removidas
        aux = inicio;
        achou = 0;
        while (aux != NULL)
        {
            if (aux->num == numero)
            {
                // o número digitado
                // foi encontrado na lista
                // e será removido
                achou = achou + 1;
```

```
if (aux == inicio)
{
    // o número a ser
    // removido
    // é o primeiro da
    // lista
    inicio = aux->prox;
    if (inicio != NULL)
    {
        inicio->ant = NULL;
    }
    delete(aux);
    aux = inicio;
}
else if (aux == fim)
{
    // o número a ser
    // removido
    // é o último da lista
    fim = fim->ant;
    fim->prox = NULL;
    delete(aux);
    aux = NULL;
}
else
{
    // o número a ser
    // removido
    // está no meio da
    // lista
    aux->ant->prox = aux-
    ↳ >prox;
    aux->prox->ant = aux-
    ↳ >ant;
```

Lista dupla enc. ordenada

```
        LISTA *aux2;
        aux2 = aux->prox;
        delete(aux);
        aux=aux2;
    }
    else
    {
        aux = aux->prox;
    }
}

if (achou == 0)
cout<<"Número não encontrado";
else if (achou == 1)
    cout<<"Número removido 1 vez";
    else
        cout<<"Número removido "<<achou<<"
        ↳ vezes";
    }
}

if (op == 5)
{
    if (inicio == NULL)
    {
        // a lista está vazia
        cout<<"Lista vazia!!";
    }

    LISTA *aux2;
    aux2 = aux->prox;
    delete(aux);
    aux=aux2;
}
else
{
    // a lista será esvaziada
    aux = inicio;
    while (aux != NULL)
    {
        inicio = inicio->prox;
        delete(aux);
        aux = inicio;
    }
    cout<<"Lista esvaziada";
}

getch();
}

while (op != 6);
}
```



Perguntas?

Bibliografia da aula

- Notas de aula do Prof. Edson L.F. Senne (UNESP/INPE) em 2010.
- ASCENCIO, A. F. G.; ARAÚJO, G. S. Estrutura de dados. Algoritmos, análise da complexidade e implementação em Java e C/C++. Pearson. 2010.