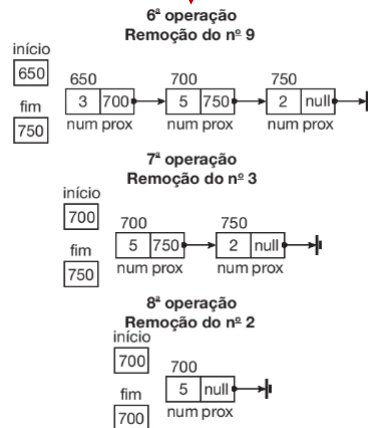
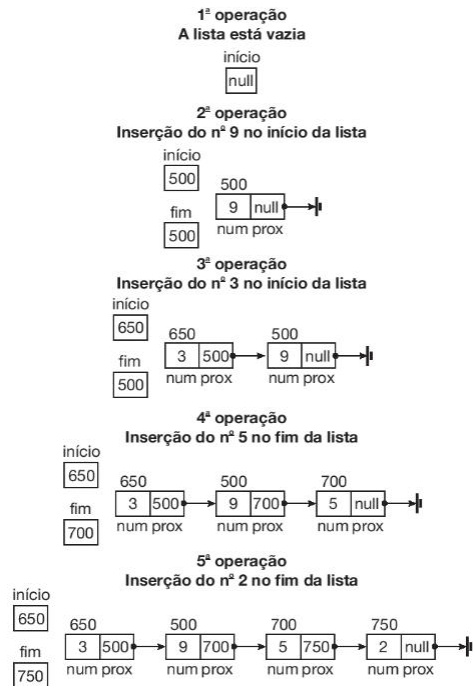


Código da lista encadeada

Murilo Dantas

Lista não ordenada



Lista não ordenada

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

void main()
{
    //Definindo o registro que representará
    //cada elemento da lista
    struct LISTA
    {
        int num;
        LISTA *prox;
    };

    // 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;
    // o ponteiro anterior é um ponteiro auxiliar
    LISTA *anterior;
    // apresentando o menu de opções
    int op, numero, achou;
```

```
do
{
    clrscr();
    cout<<"\nMENU DE OPÇÕES\n";
    cout<<"\n1 - Inserir no início da
    ↳ lista";
    cout<<"\n2 - Inserir no fim da lista";
    cout<<"\n3 - Consultar toda a lista";
    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 no início da 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
            inicio = novo;
            fim = novo;
            fim->prox = NULL;
        }
        else
        {
```

Lista não ordenada

```
// a lista já contém elementos
// e o novo elemento
// será inserido no início da
// lista
novo->prox = inicio;
inicio = novo;
}
cout<<"Número inserido no
↳ início da lista!!";
}
if (op == 2)
{
    cout<<"Digite o número a ser
    ↳ inserido no fim da 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
        inicio = novo;
        fim = novo;
        fim->prox = NULL;
    }
}
```

```
else
{
    // a lista já contém elementos e
    // o novo elemento
    // será inserido no fim da lista
    fim->prox = novo;
    fim = novo;
    fim->prox=NULL;
}
cout<<"Número inserido no fim da
↳ lista!!";
}
if (op == 3)
{
    if (inicio == NULL)
    {
        // a lista está vazia
        cout<<"Lista vazia!!";
    }
    else
    {
```

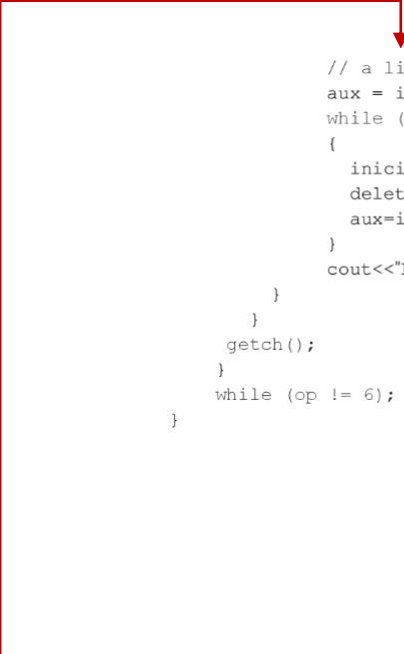
Lista não ordenada

```
// a lista contém elementos e estes serão
// mostrados do início ao fim
cout<<"\nConsultando toda a lista\n";
aux = inicio;
while (aux != NULL)
{
    cout<<aux->num<<" ";
    aux = aux->prox;
}
}
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;
        anterior = NULL;
        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;
            delete(aux);
            aux = inicio;
        }
    }
    else if (aux == fim)
    {
        // o número a ser removido
        // é o último da lista
        anterior->prox = NULL;
        fim = anterior;
        delete(aux);
        aux = NULL;
    }
    else
    {
        // o número a ser
        // removido
        // está no meio da
        // lista
        anterior->prox = aux->prox;
        delete(aux);
        aux = anterior->prox;
    }
}
```

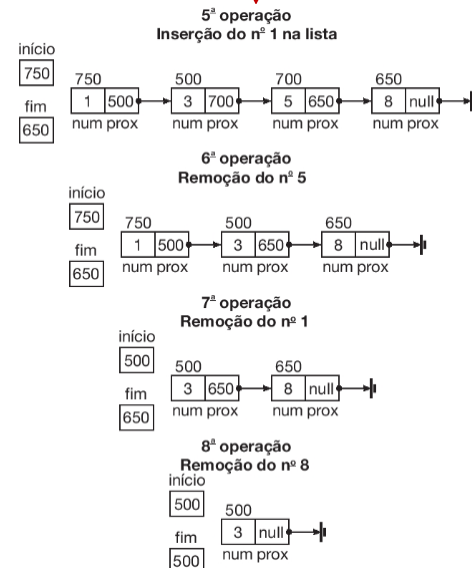
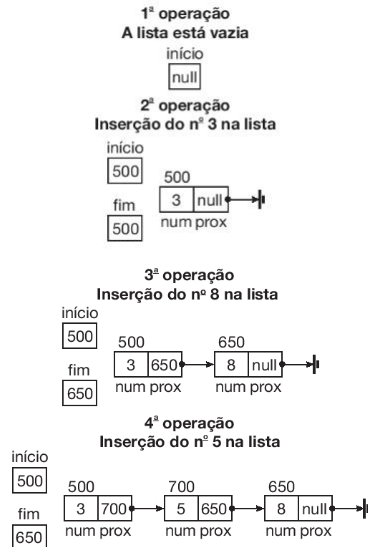
Lista não ordenada

```
        else
        {
            anterior = aux;
            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!";
    }
    else
    {
```



```
        // a lista será esvaziada
        aux = inicio;
        while (aux != NULL)
        {
            inicio = inicio->prox;
            delete(aux);
            aux=inicio;
        }
        cout<<"Lista esvaziada";
    }
    getch();
}
while (op != 6);
}
```

Lista ordenada



Lista ordenada

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

void main()
{
    //Definindo o registro que
    //representará cada elemento da lista
    struct LISTA
    {
        int num;
        LISTA *prox;
    };

    // a lista está vazia, logo,
    // o ponteiro inicio têm 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;
    // o ponteiro anterior é um ponteiro auxiliar
    LISTA *anterior;
    // apresentando o menu de opções
    int op, numero, achou;
    do
```

```
{
    clrscr();
    cout<<"\nMENU DE OPÇÕES\n";
    cout<<"\n1 - Inserir na lista";
    cout<<"\n2 - Consultar toda a lista";
    cout<<"\n3 - Remover da lista";
    cout<<"\n4 - Esvaziar a lista";
    cout<<"\n5 - Sair";
    cout<<"\nDigite sua opção: ";
    cin>>op;
    if (op < 1 || op > 5)
        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
            inicio = novo;
            fim = novo;
            novo->prox = NULL;
        }
    }
}
```


Lista ordenada

```
else
{
    // a lista já contém elementos
    // e o novo elemento
    // será inserido na lista
    // respeitando a ordenação
    // crescente
    anterior = NULL;
    aux = inicio;
    while (aux != NULL
    && novo->num > aux->num)
    {
        anterior = aux;
        aux = aux->prox;
    }
    if (anterior == NULL)
    {
        // o novo número a ser inserido
        // é menor que todos os
        // números da lista,
        // logo, será inserido no
        // início
        novo->prox = inicio;
        inicio = novo;
    }
    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;
        fim = novo;
        fim->prox=NULL;
    }
    else
    {
        // o novo número a ser
        // inserido
        // será inserido entre
        // dois
        // números que já estão na
        // lista
        anterior->prox =
        ~ novo;
        novo->prox = aux;
    }
    }
    cout<<"Número inserido na lista!!";
}
if (op == 2)
{
    if (inicio == NULL)
    {
        // a lista está vazia
        cout<<"Lista vazia!!";
```

Lista ordenada

```
}
else
{
    // a lista contém elementos e
    // estes serão
    // mostrados do início ao fim
    cout<<"\nConsultando toda a
    ↳ lista\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 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;
anterior = NULL;
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;
            delete(aux);
            aux = inicio;
        }
        else if (aux == fim)
        {
            // o número a ser
            // removido
            // é o último da
            // lista
            anterior->prox =
            ↳ NULL;
            fim = anterior;
            delete(aux);
            aux = NULL;
        }
    }
}
```

Lista ordenada

```
        else
        {
            // o número a ser
            // removido
            // está no meio
            // da lista
            anterior->prox
            ↳ =aux->prox;
            delete(aux);
            aux = anterior->
            ↳ prox;
        }
    }
    else
    {
        anterior = aux;
        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 == 4)
{
    if (inicio == NULL)
    {
        // a lista está vazia
        cout<<"Lista vazia!!";
    }
}
```



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

    }
    getch();
}
while (op != 5);
}
```

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.