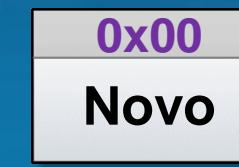


Tecnologia Em Analise e Desenv. de Sistemas

Estrutura de Dados

Aula 09 – Lista Duplamente Encadeada

Prof. Dr. Ricardo Luis Balieiro



Lista dinâmica duplamente encadeada

- Neste tipo de lista a memória para armazenamento de cada elemento é alocada em tempo de execução utilizando ponteiros.
- Cada elemento da lista dinâmica é chamado de **NÓ**.
- O nó é composto por duas partes:
 - **Dados**: dados que serão armazenados
 - **Próximo**: aponta para o próximo nó
 - **Anterior**: aponta para o nó anterior



INSERIR NO INÍCIO

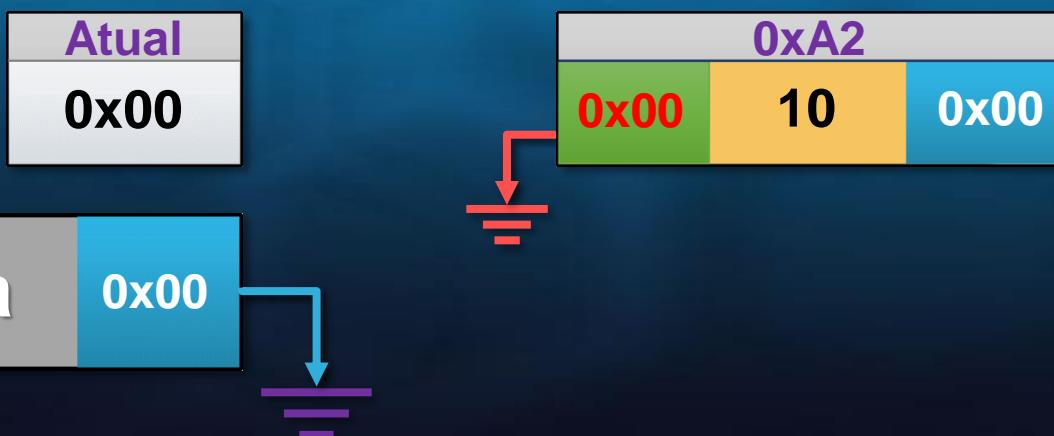
SIMULAÇÃO 1

Inserir no início – Simulação 1



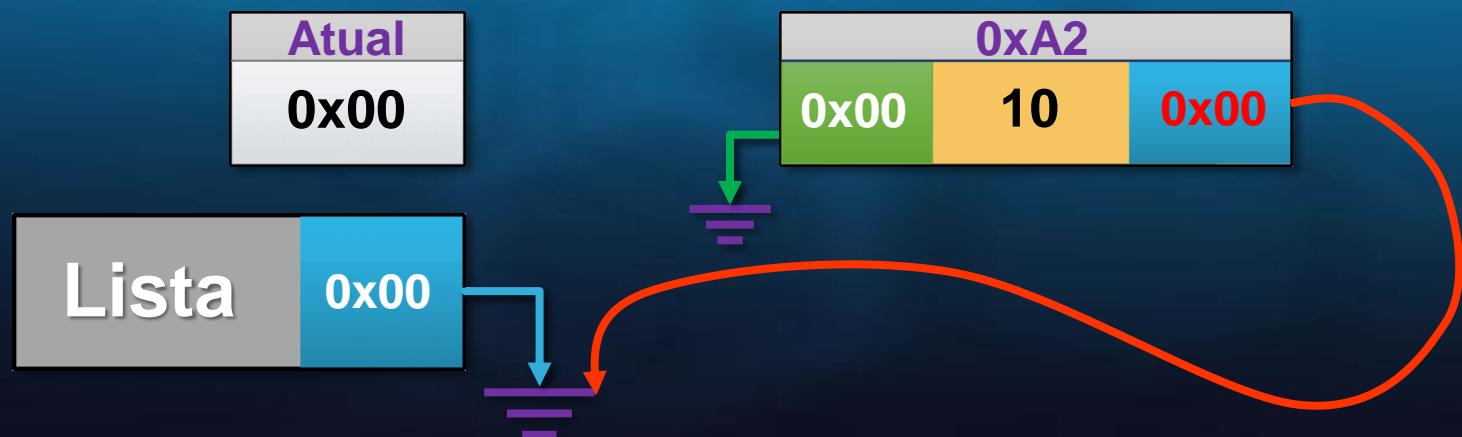
Inserir no início – Simulação 1

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



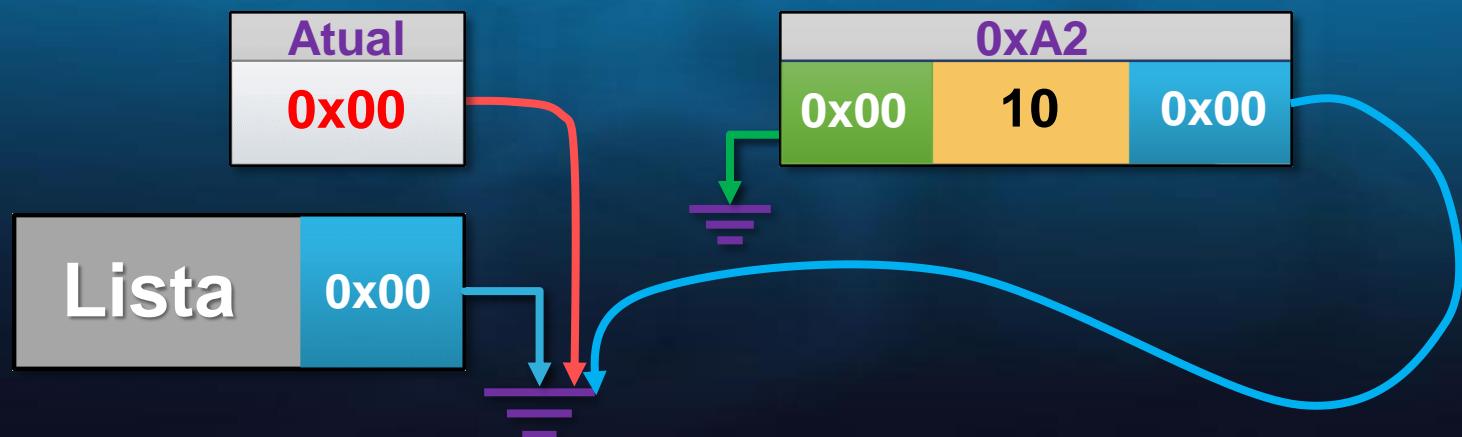
Inserir no início – Simulação 1

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



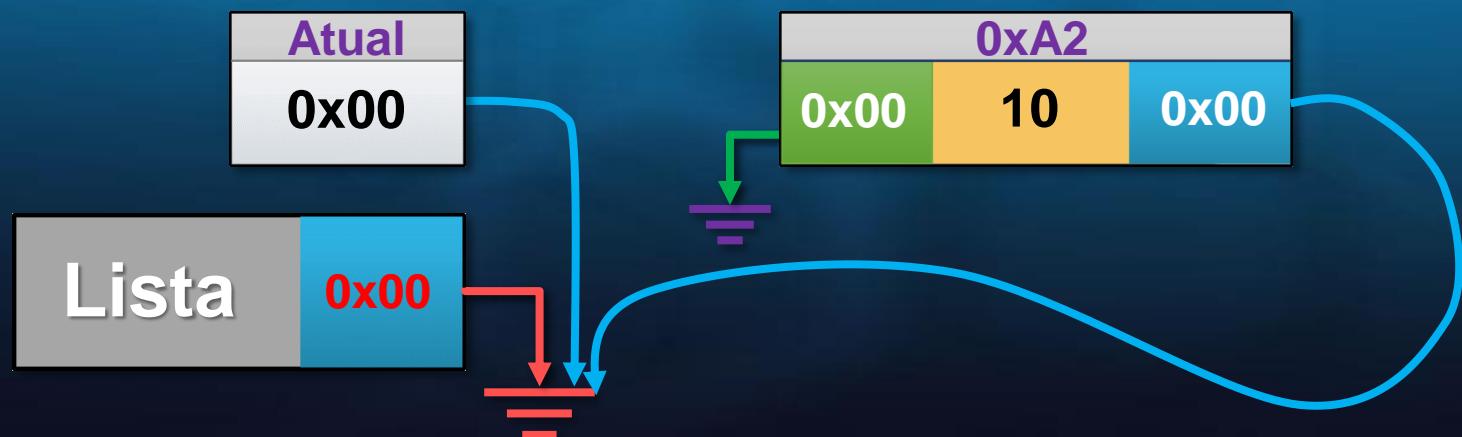
Inserir no início – Simulação 1

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



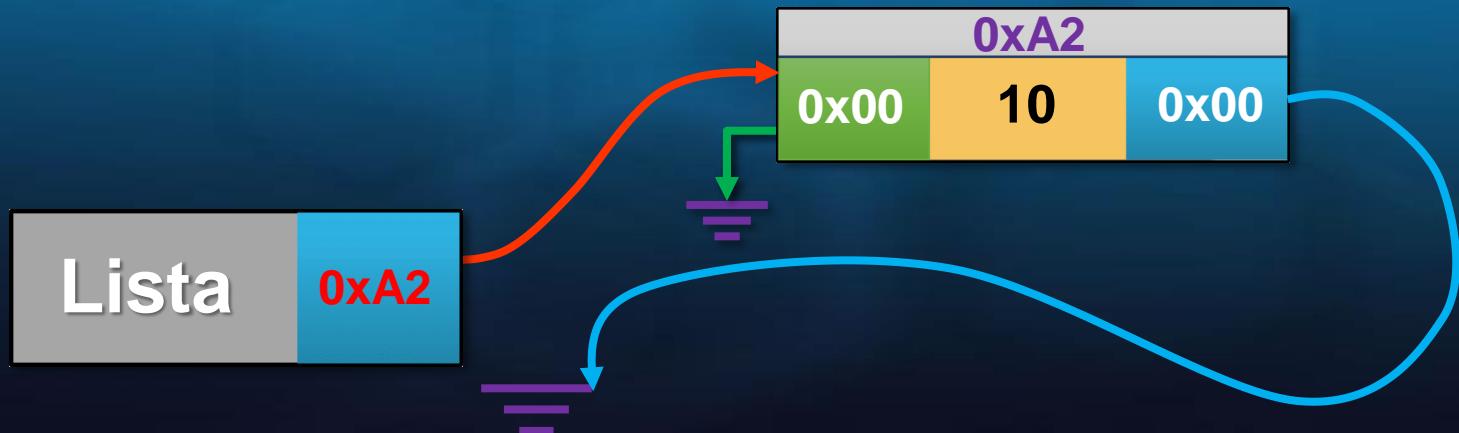
Inserir no início – Simulação 1

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



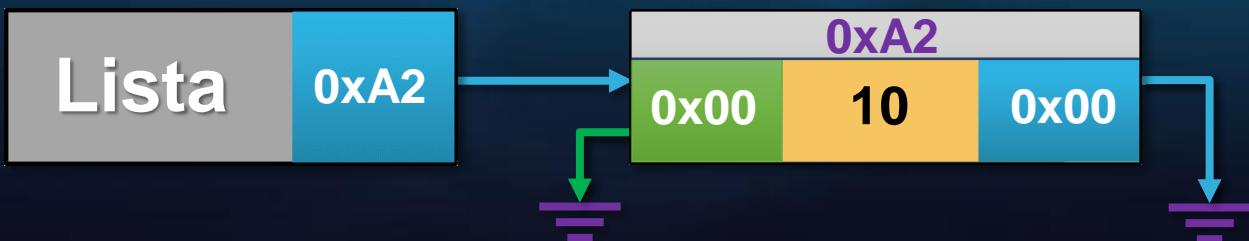
Inserir no início – Simulação 1

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



Inserir no início – Simulação 1

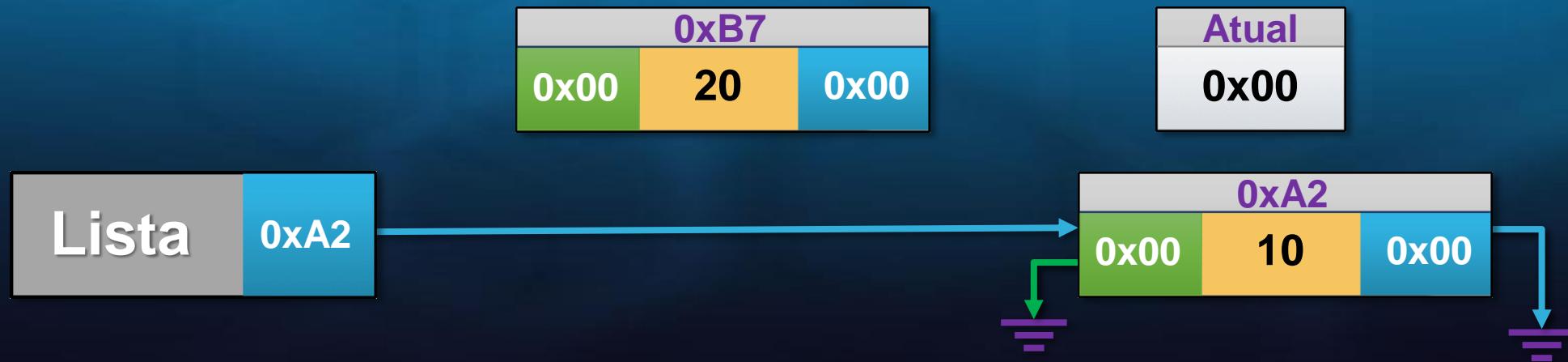
```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



INSERIR NO INÍCIO

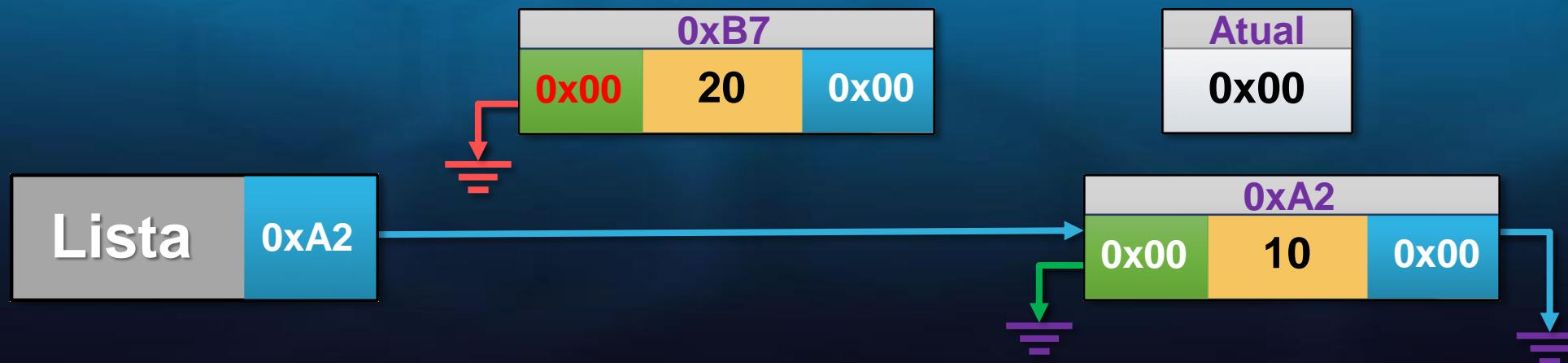
SIMULAÇÃO 2

Inserir no início – Simulação 2



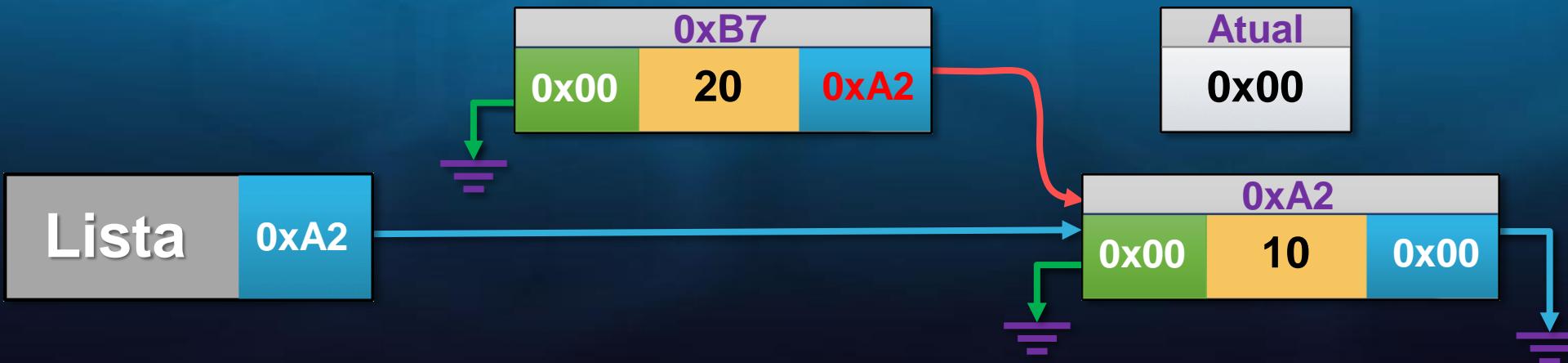
Inserir no início – Simulação 2

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



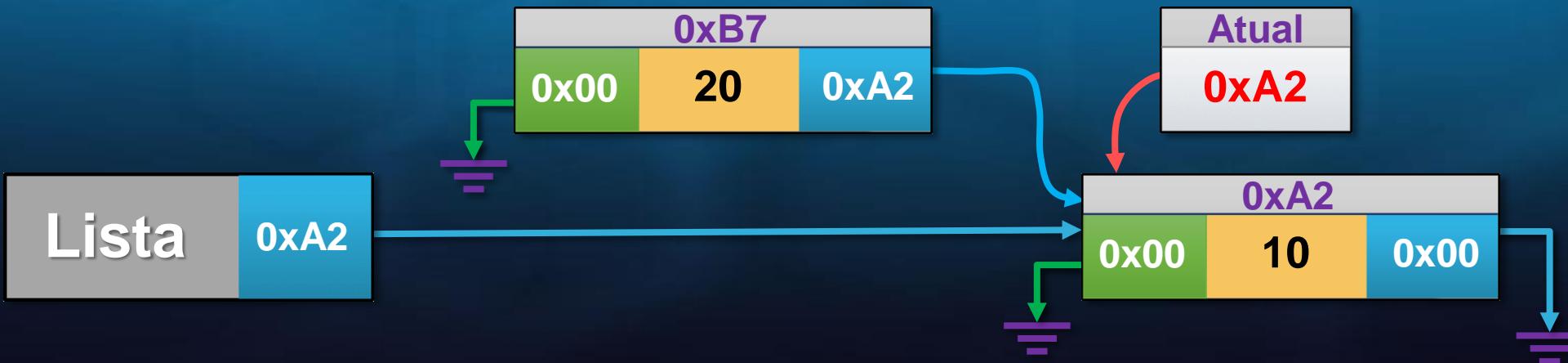
Inserir no início – Simulação 2

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



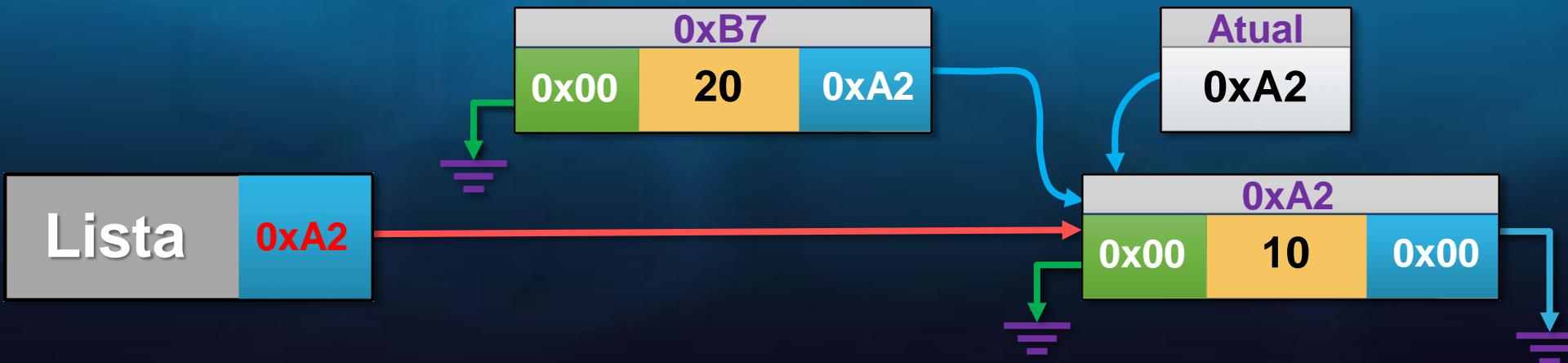
Inserir no início – Simulação 2

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



Inserir no início – Simulação 2

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



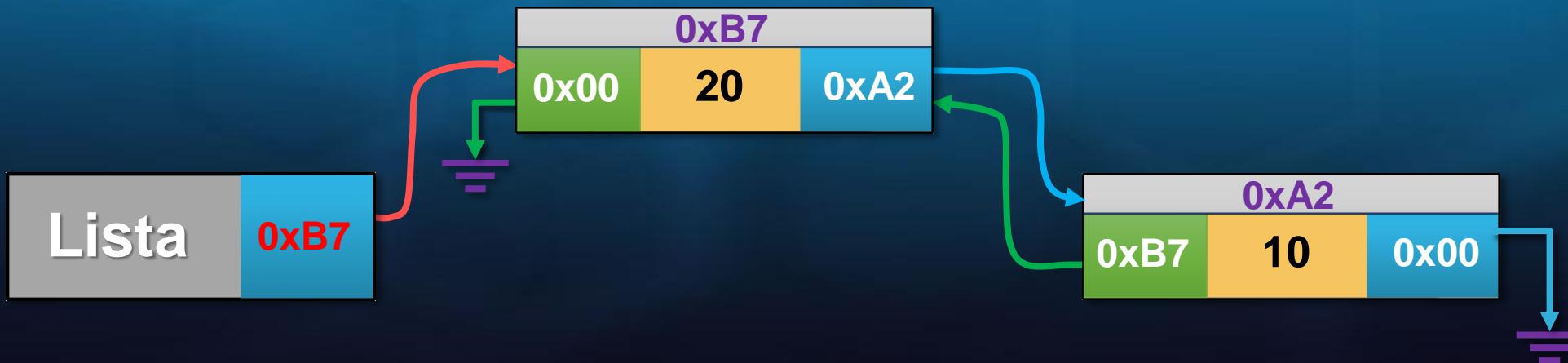
Inserir no início – Simulação 2

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



Inserir no início – Simulação 2

```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



Inserir no início – Simulação 2

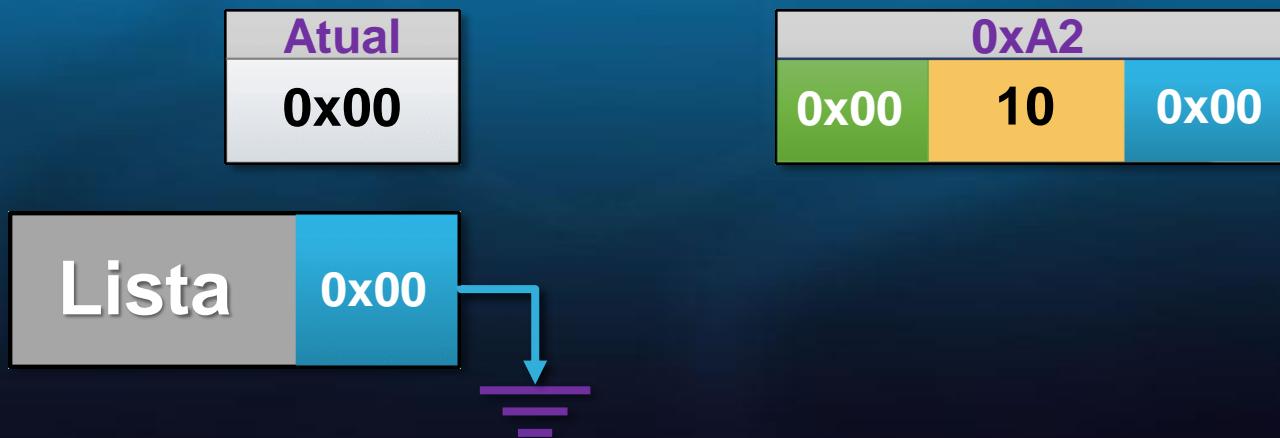
```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = ptrLista->inicio;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se a lista não estiver vazia  
if (ptrLista->inicio != NULL) {  
    ptrNoAtual->AntNo = ptrNoNovo;  
}  
  
ptrLista->inicio = ptrNoNovo;
```



INSERIR NO FINAL

SIMULAÇÃO 1

Inserir no final – Simulação 1



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

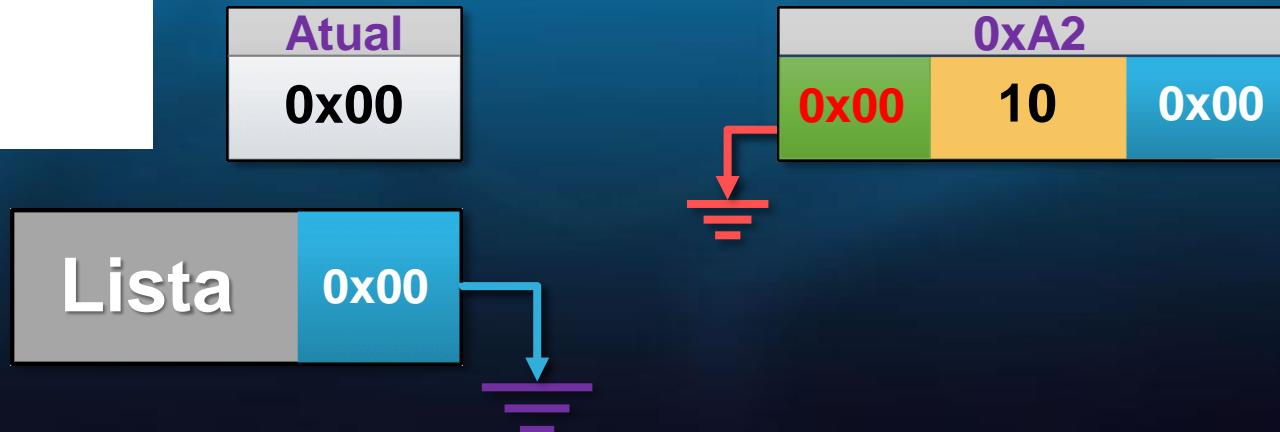
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

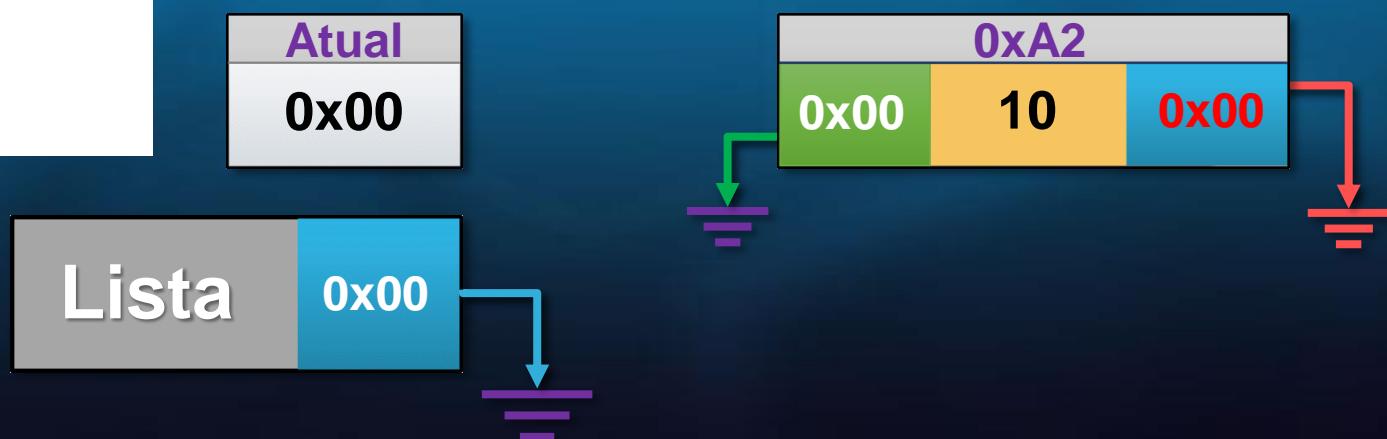
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

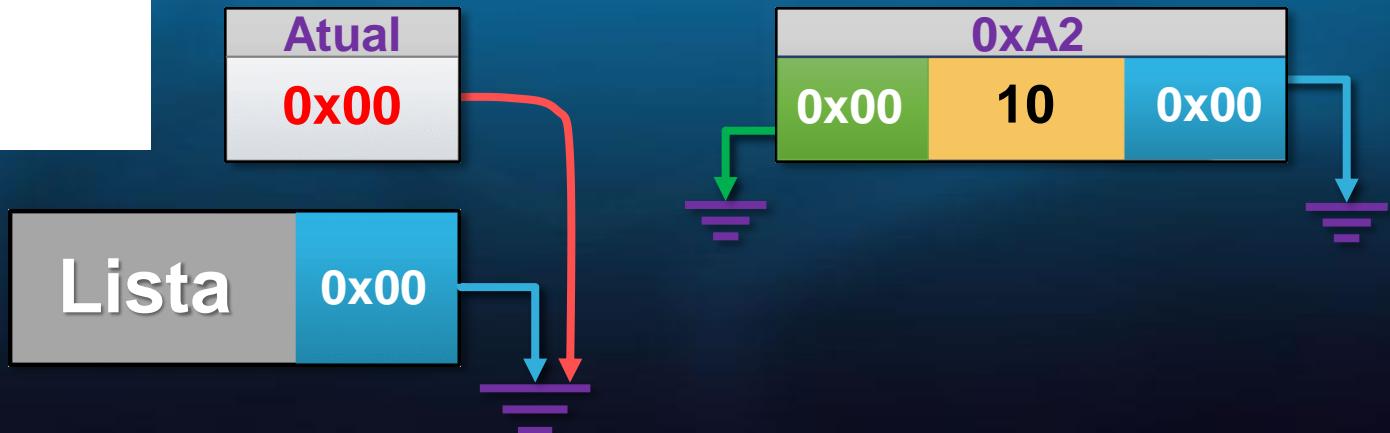
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

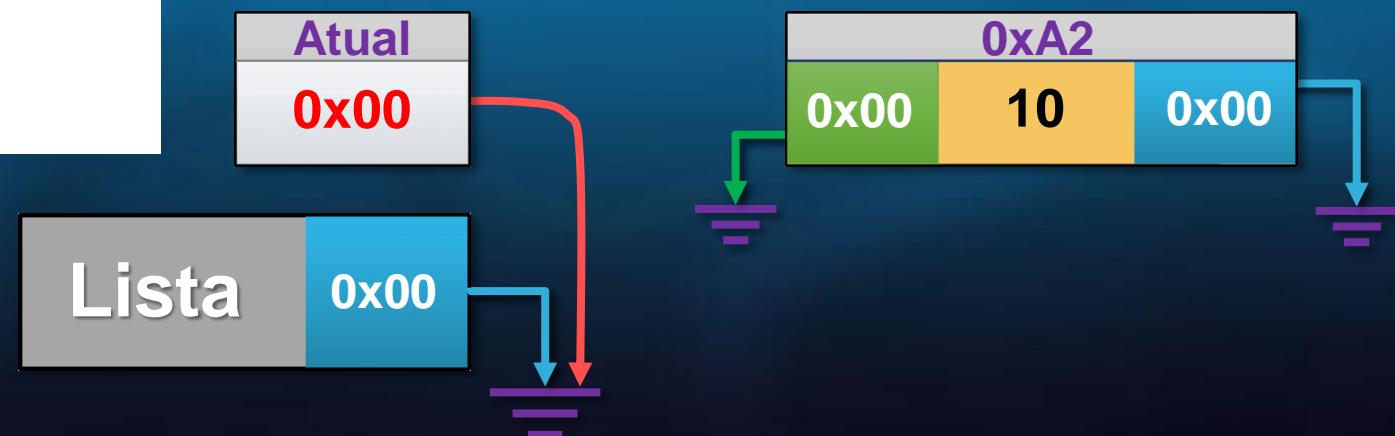
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

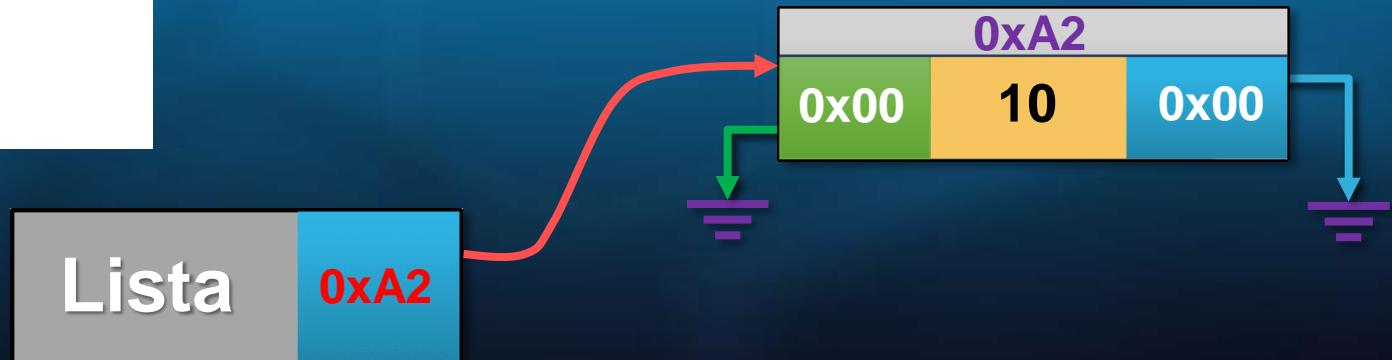
    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

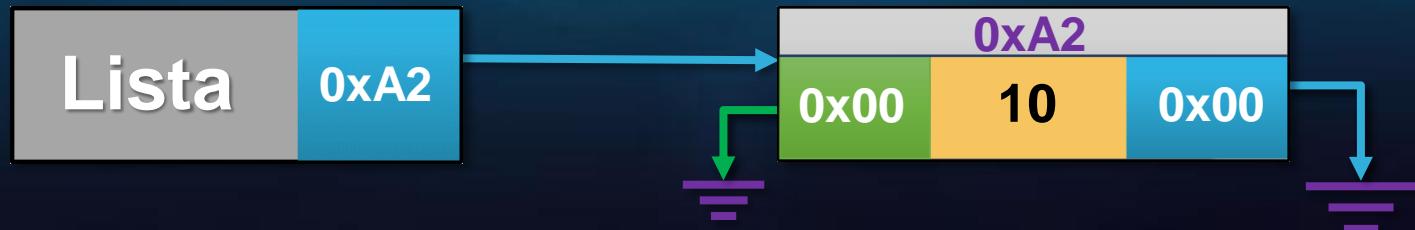
```



```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se não houver nenhum nó na lista  
if (ptrNoAtual == NULL) {  
  
    ptrLista->inicio = ptrNoNovo;  
}  
else {  
  
    // Localiza o último nó  
    while (ptrNoAtual->proxNo != NULL) {  
        ptrNoAtual = ptrNoAtual->proxNo;  
    }  
    ptrNoAtual->proxNo = ptrNoNovo;  
    ptrNoNovo->AntNo = ptrNoAtual;  
}
```



```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se não houver nenhum nó na lista  
if (ptrNoAtual == NULL) {  
  
    ptrLista->inicio = ptrNoNovo;  
}  
else {  
  
    // Localiza o último nó  
    while (ptrNoAtual->proxNo != NULL) {  
        ptrNoAtual = ptrNoAtual->proxNo;  
    }  
    ptrNoAtual->proxNo = ptrNoNovo;  
    ptrNoNovo->AntNo = ptrNoAtual;  
}
```



INserir no final

Simulação 2

Inserir no final – Simulação 2



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

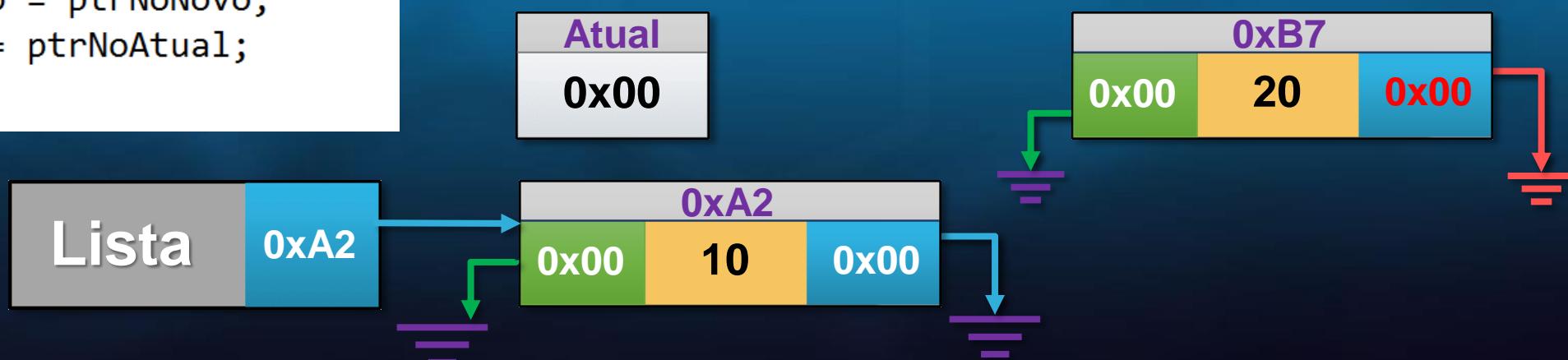
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

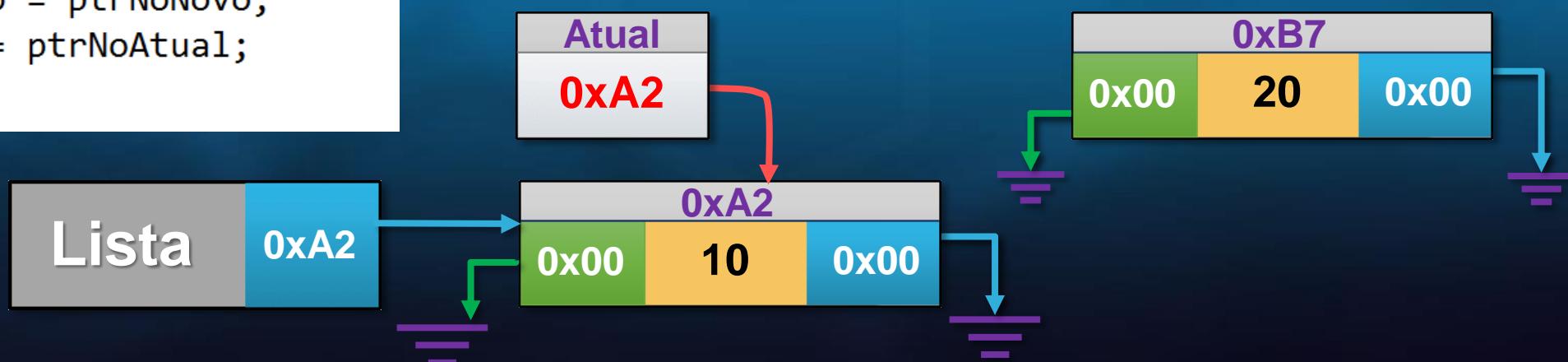
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

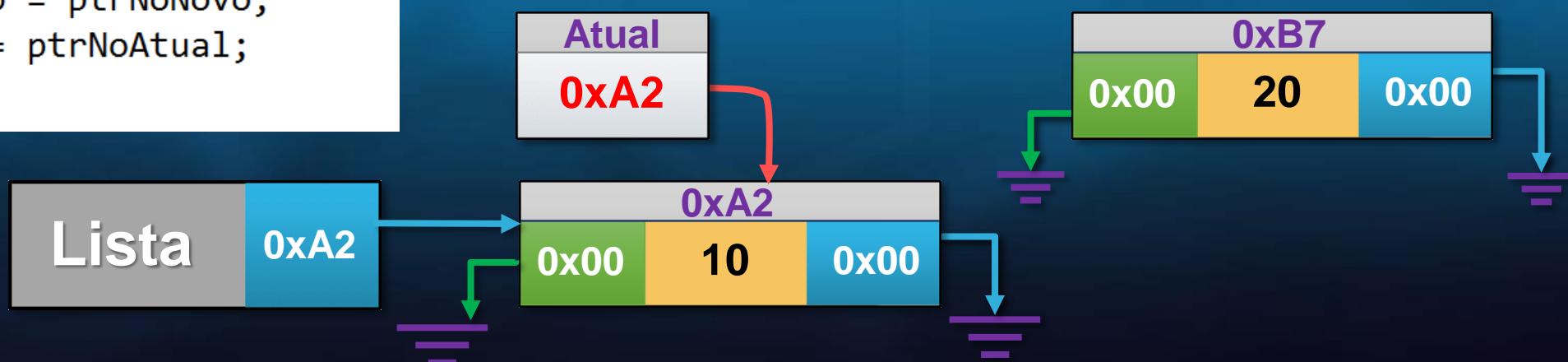
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

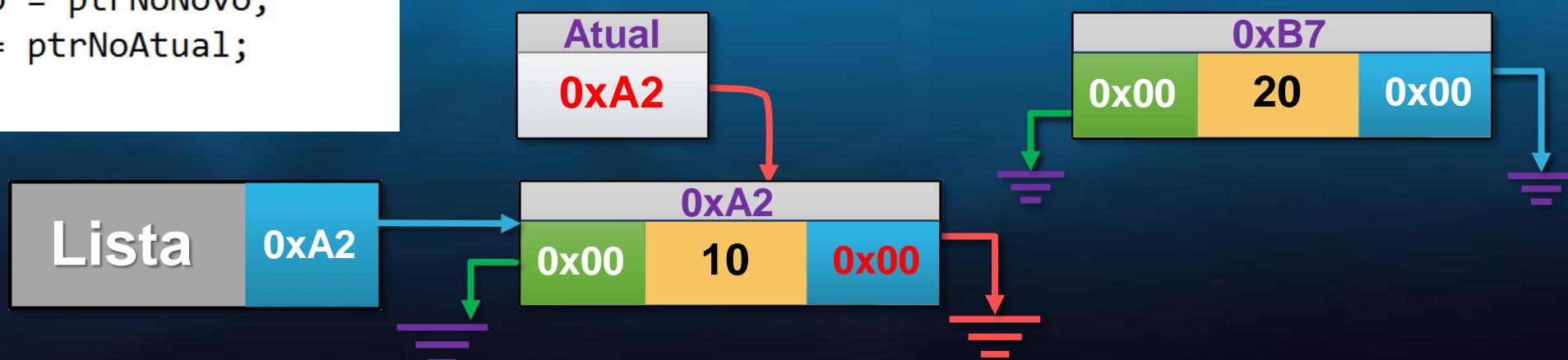
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

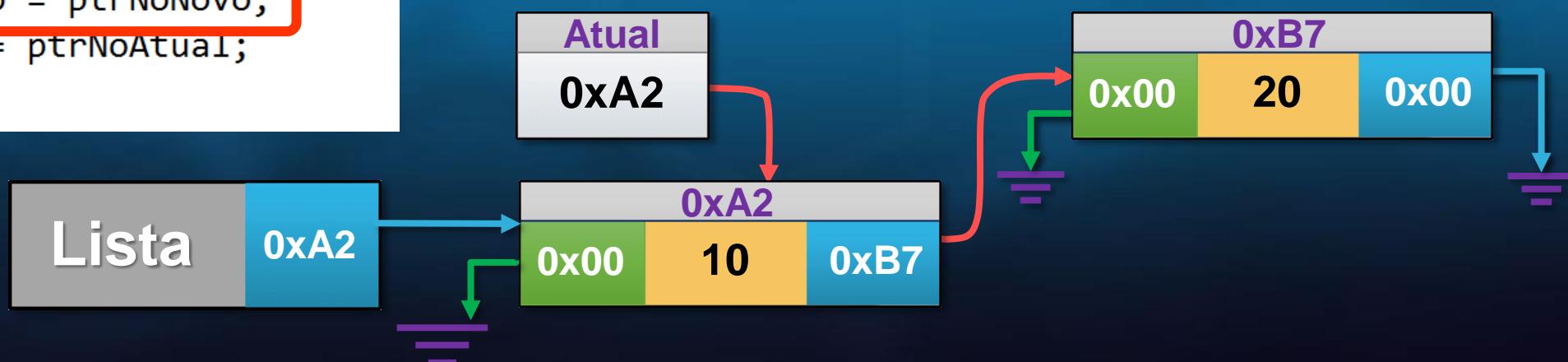
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

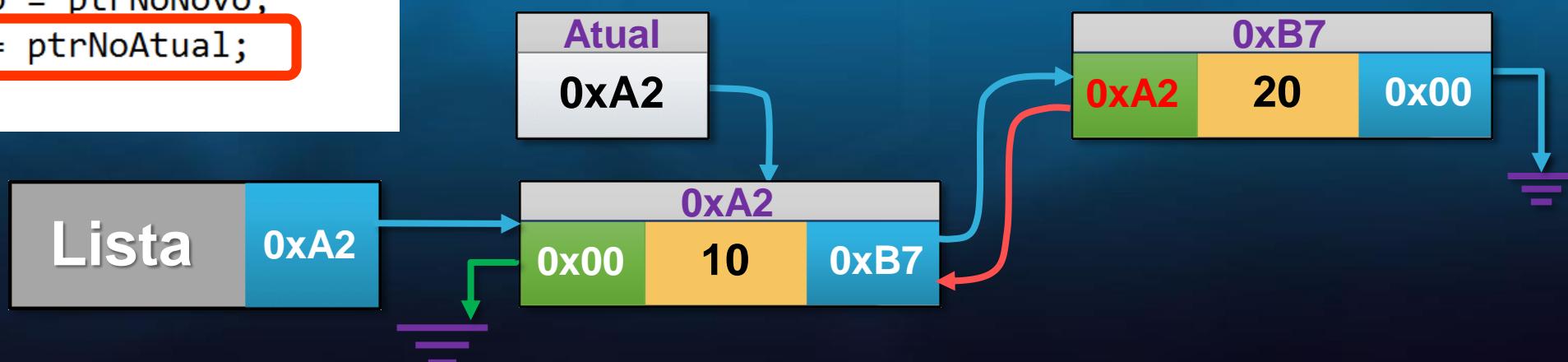
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

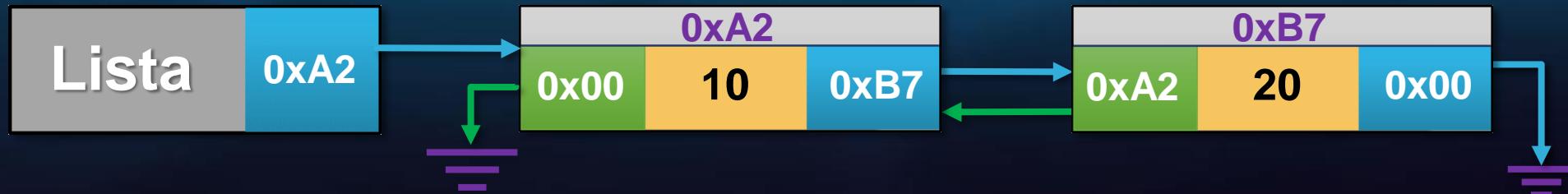
    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



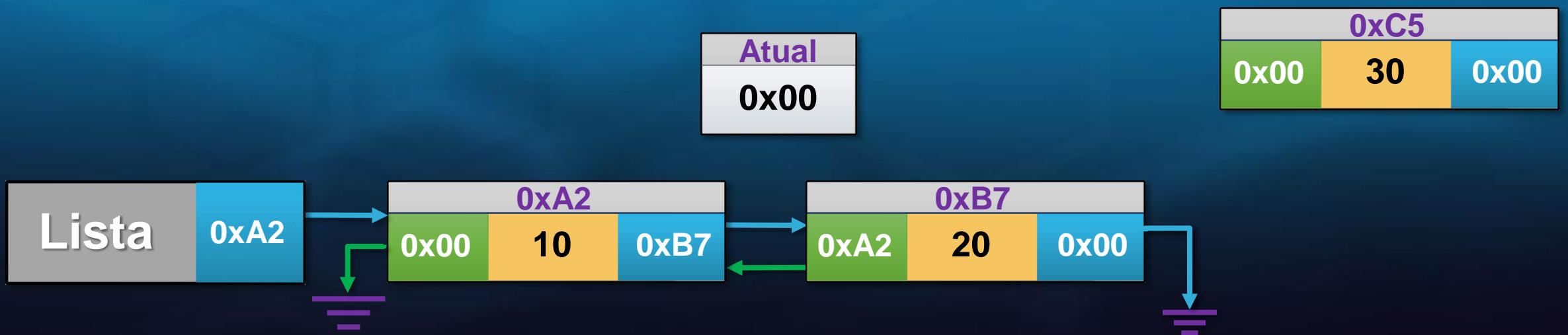
```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;  
  
ptrNoAtual = ptrLista->inicio;  
  
// Se não houver nenhum nó na lista  
if (ptrNoAtual == NULL) {  
  
    ptrLista->inicio = ptrNoNovo;  
}  
else {  
  
    // Localiza o último nó  
    while (ptrNoAtual->proxNo != NULL) {  
        ptrNoAtual = ptrNoAtual->proxNo;  
    }  
    ptrNoAtual->proxNo = ptrNoNovo;  
    ptrNoNovo->AntNo = ptrNoAtual;  
}
```



INserir no final

Simulação 3

Inserir no final – Simulação 3



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

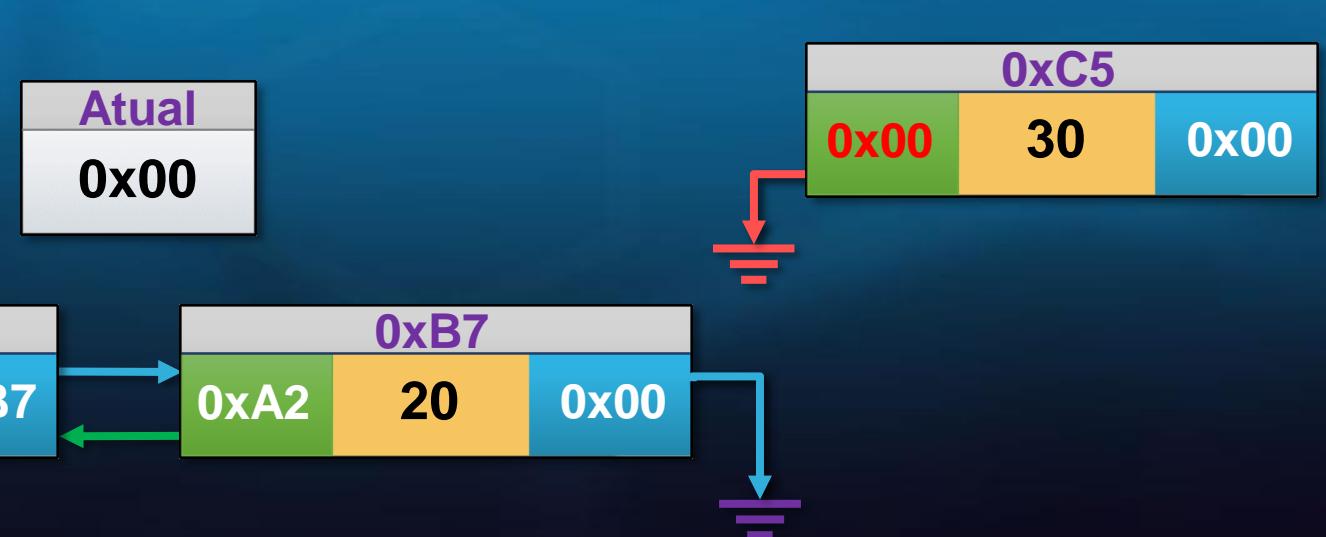
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

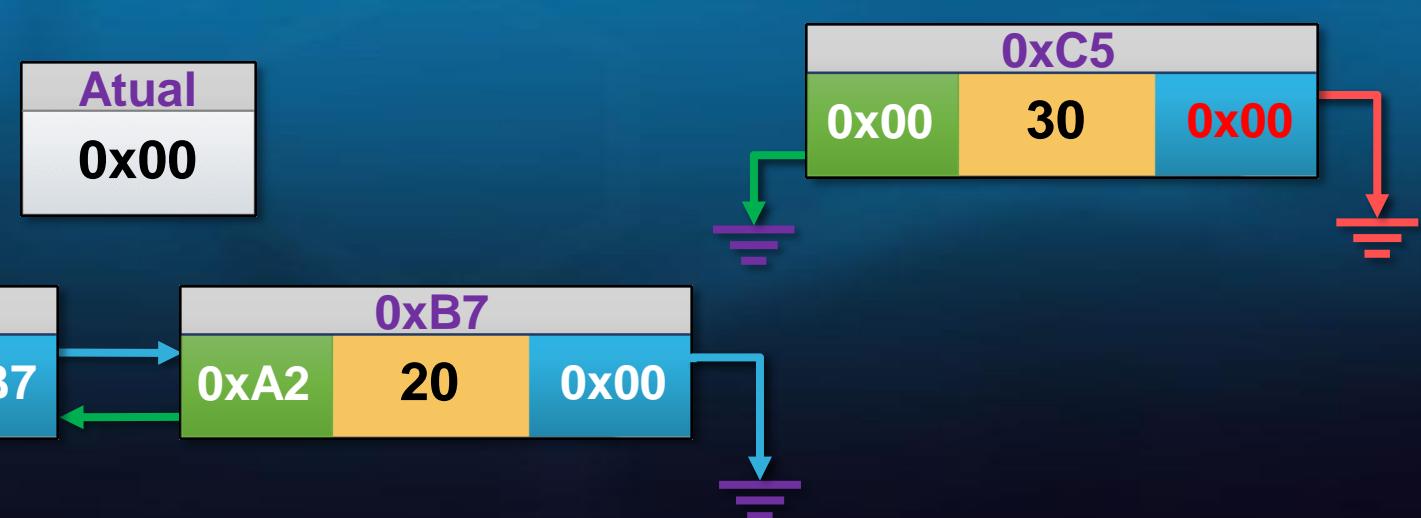
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

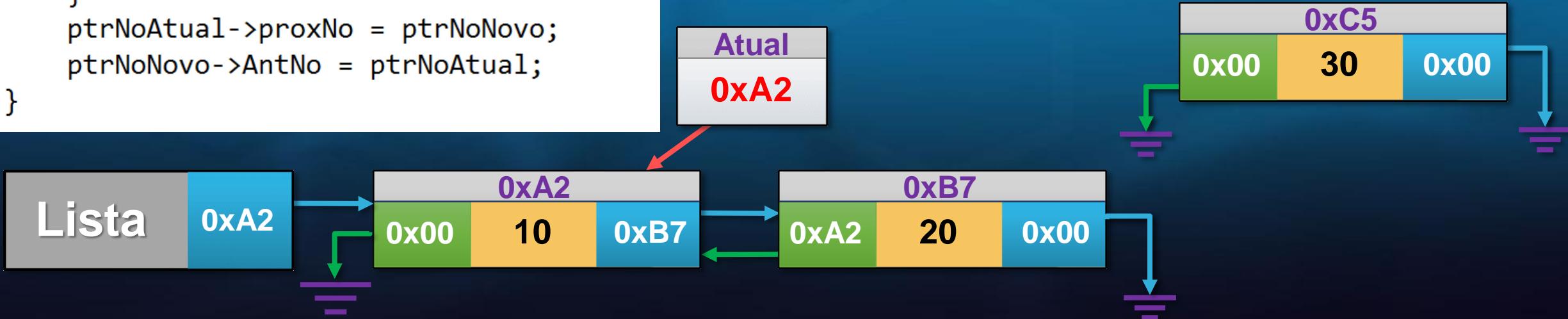
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

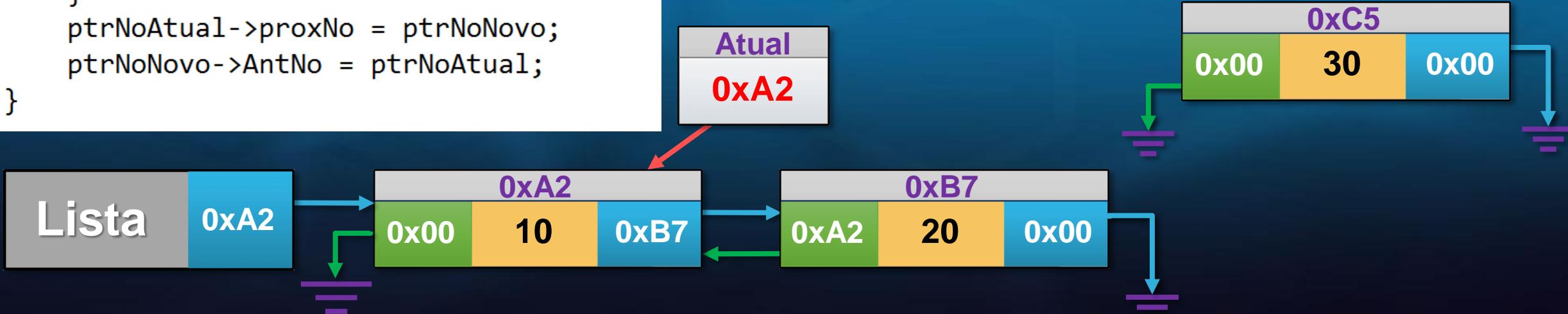
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

ptrNoAtual = ptrLista->inicio;

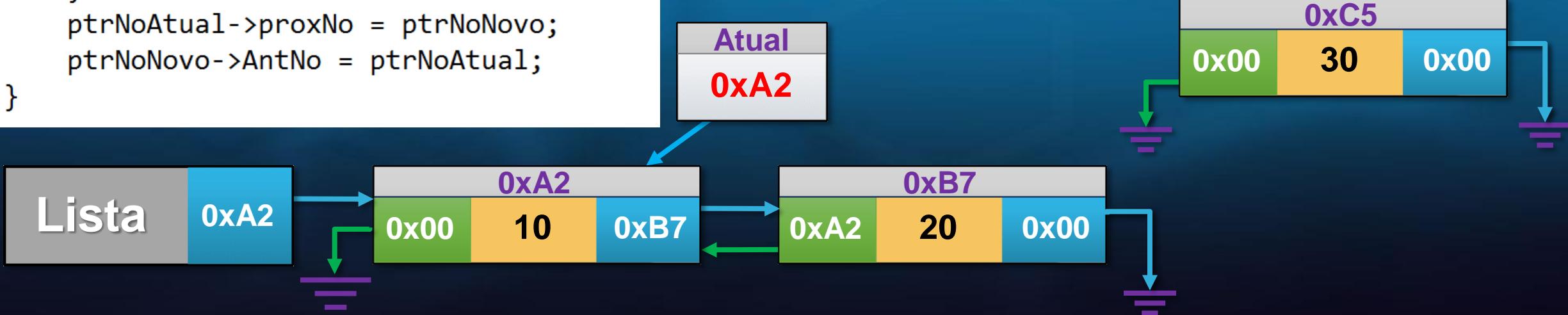
// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}

else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

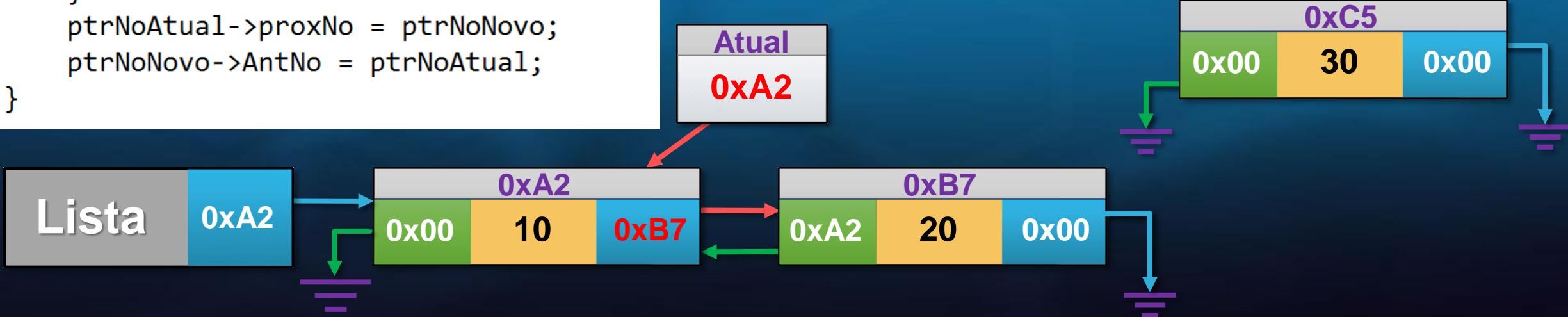
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

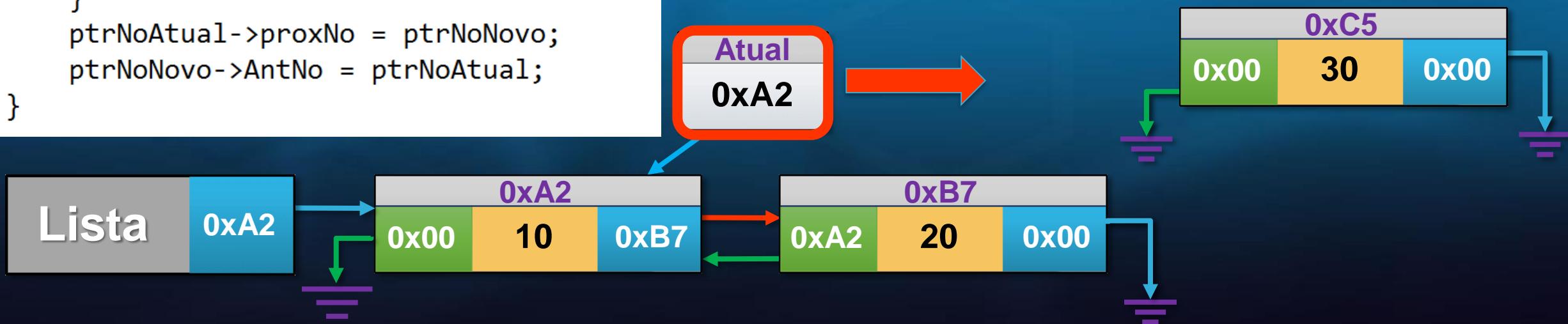
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

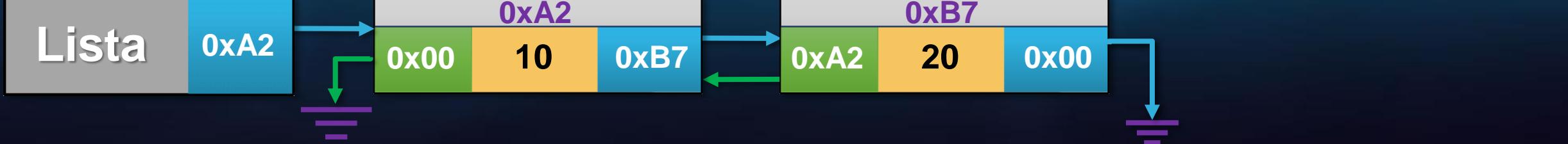
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

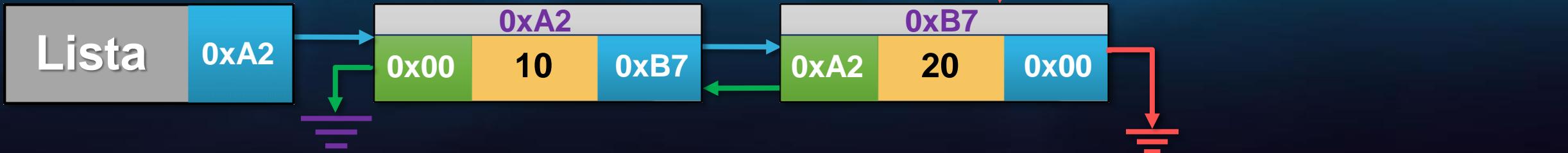
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

ptrNoAtual = ptrLista->inicio;

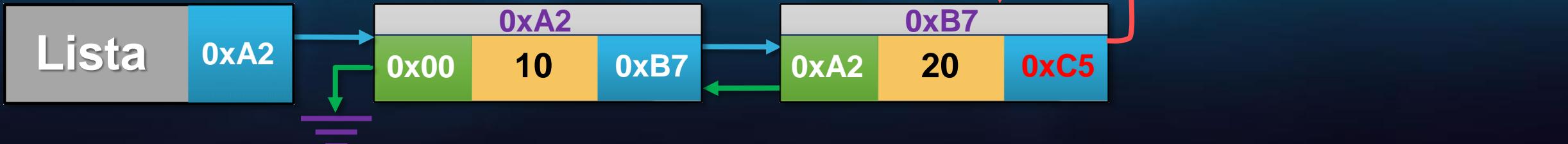
// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }

    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

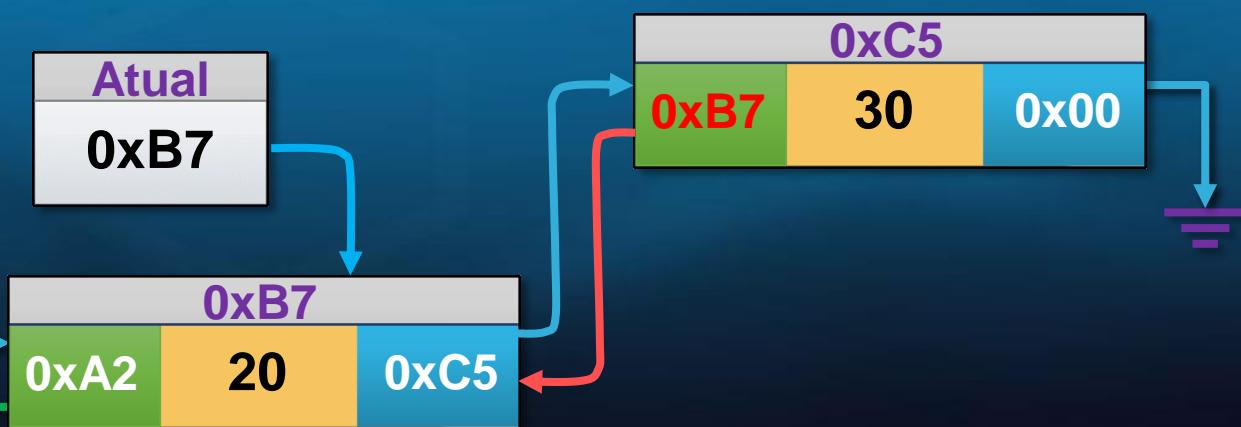
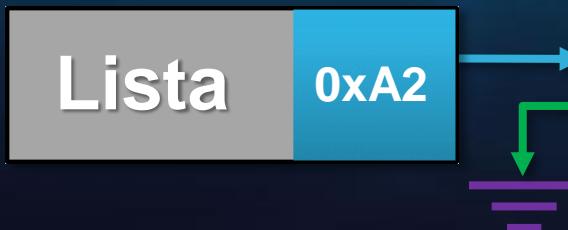
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

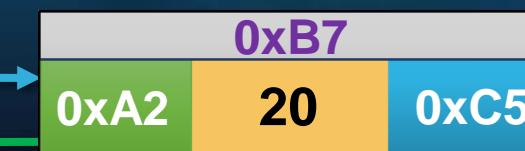
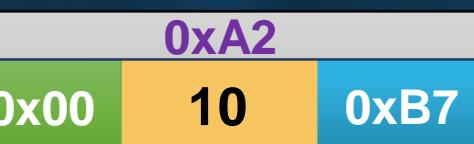
    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```

Lista

0xA2



```

ptrNoNovo->AntNo = NULL;
ptrNoNovo->proxNo = NULL;

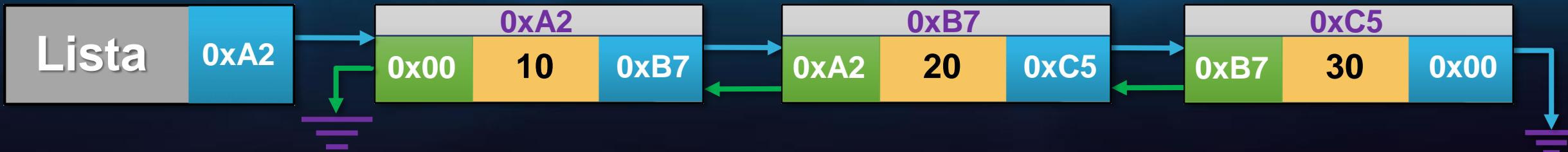
ptrNoAtual = ptrLista->inicio;

// Se não houver nenhum nó na lista
if (ptrNoAtual == NULL) {

    ptrLista->inicio = ptrNoNovo;
}
else {

    // Localiza o último nó
    while (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual = ptrNoAtual->proxNo;
    }
    ptrNoAtual->proxNo = ptrNoNovo;
    ptrNoNovo->AntNo = ptrNoAtual;
}

```

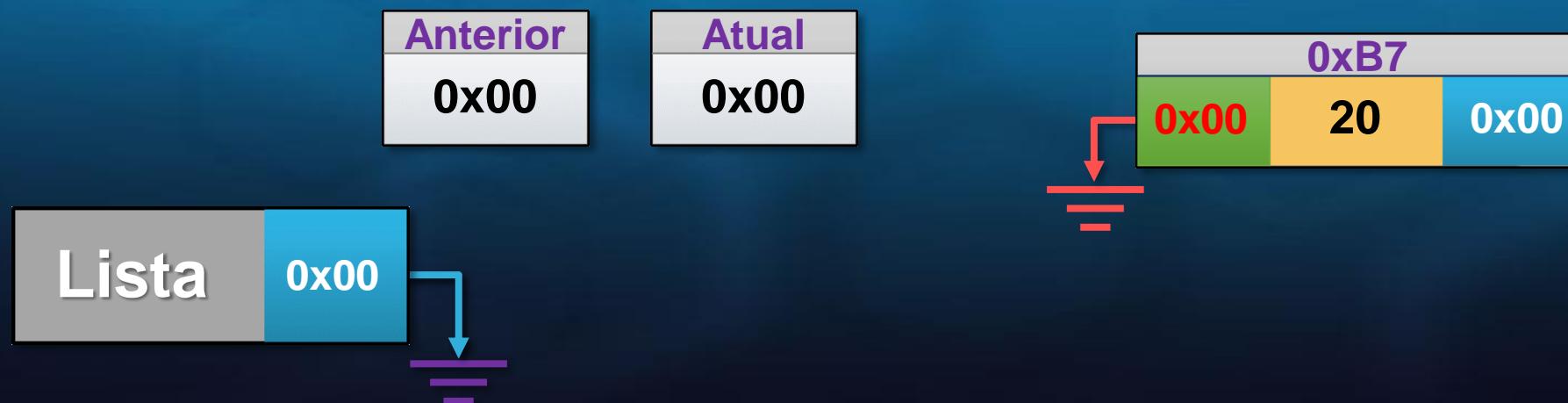


INSERIR ORDENADO NA LISTA

SIMULAÇÃO 1



```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;
```



```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;
```



```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```



```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

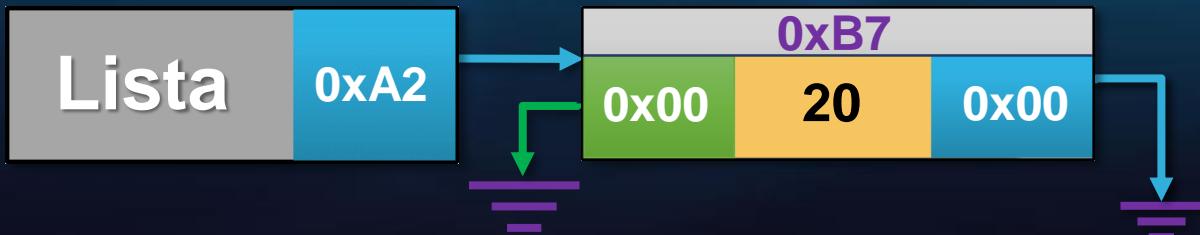
    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```



```
//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}
```



INSERIR ORDENADO NA LISTA

SIMULAÇÃO 2

Anterior
0x00

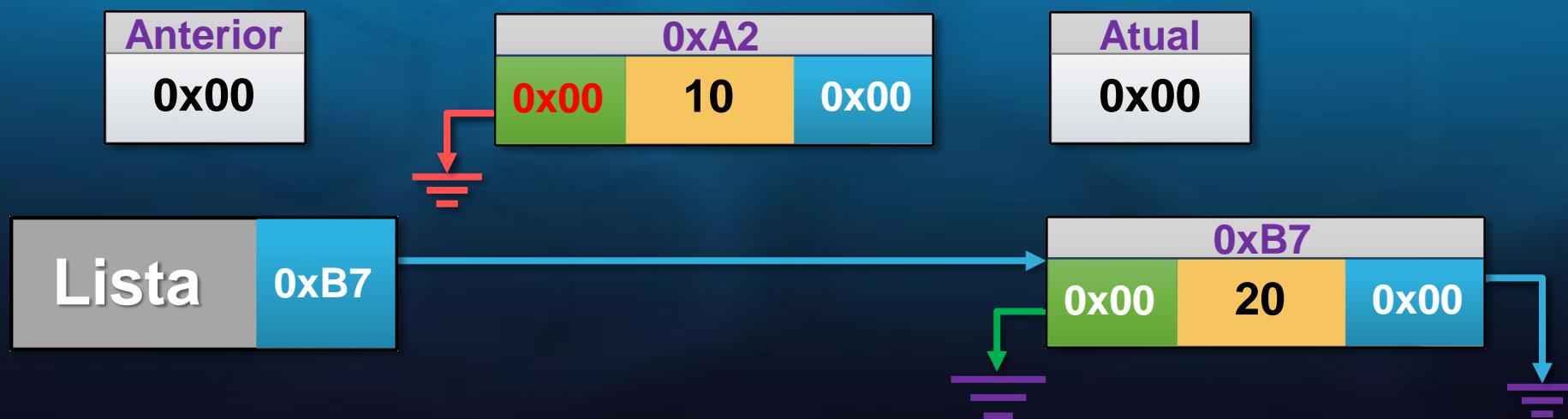
0xA2
0x00 10 0x00

Atual
0x00

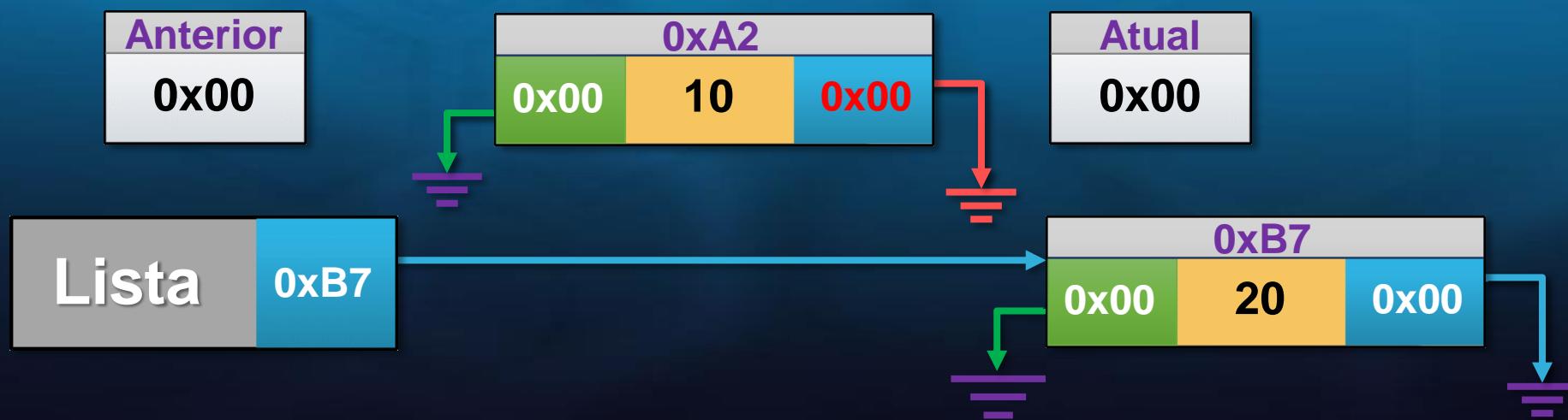
Lista 0xB7



```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;
```



```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;
```

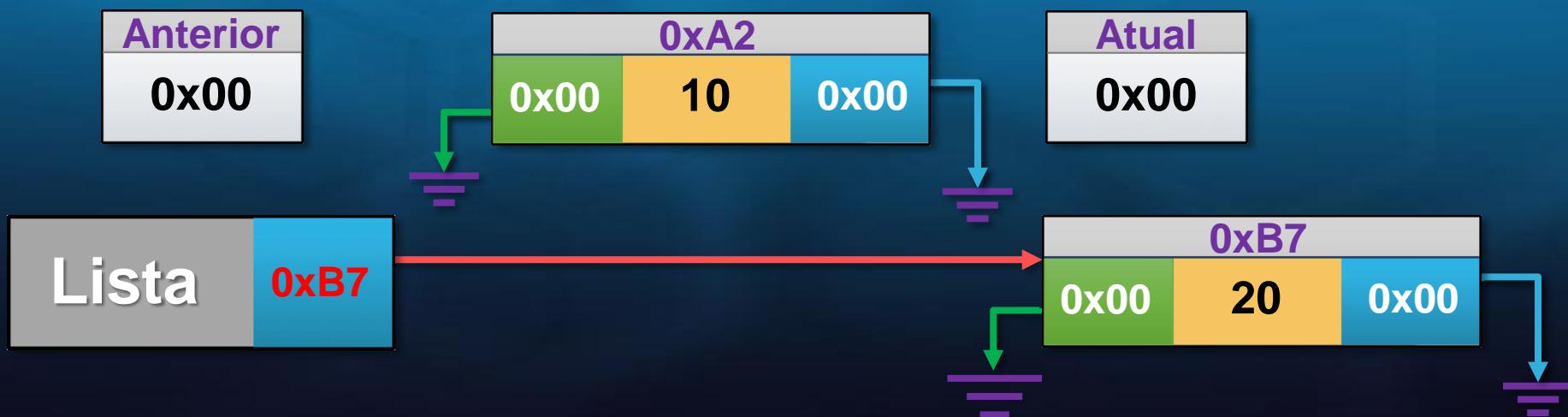


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

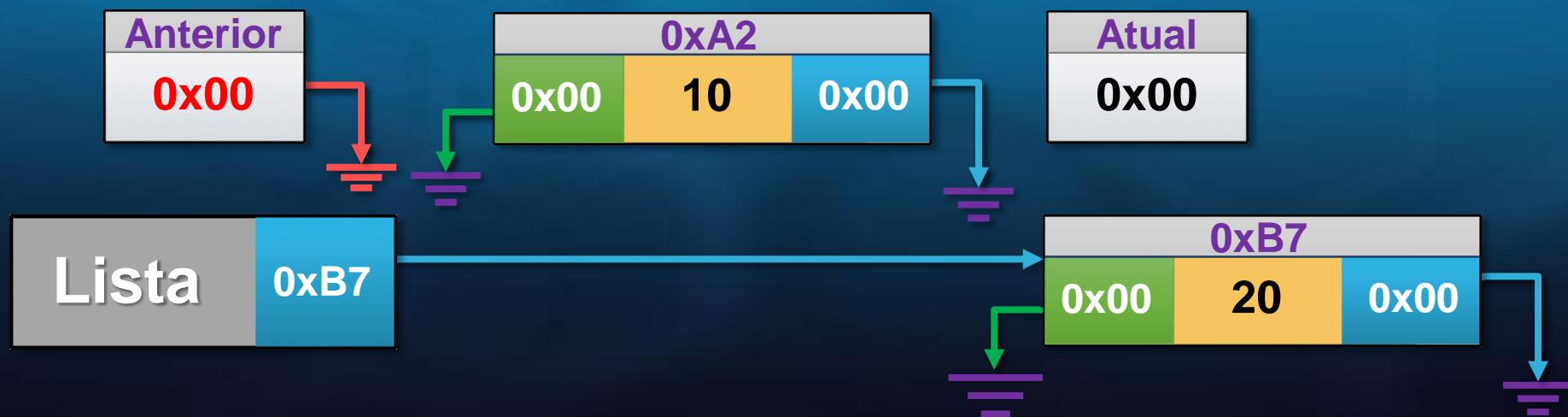


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

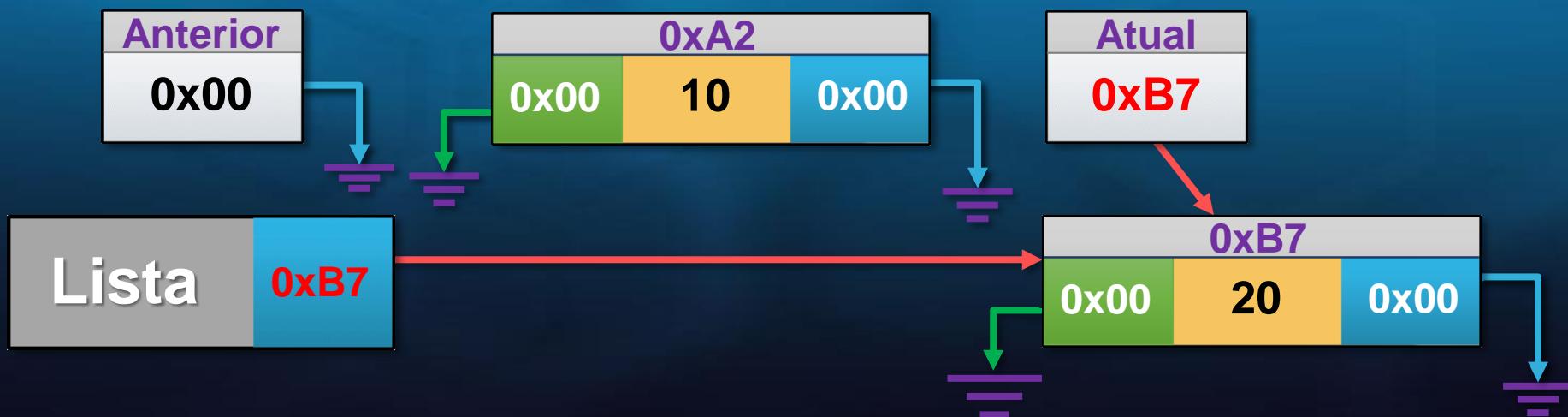


```

// Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```



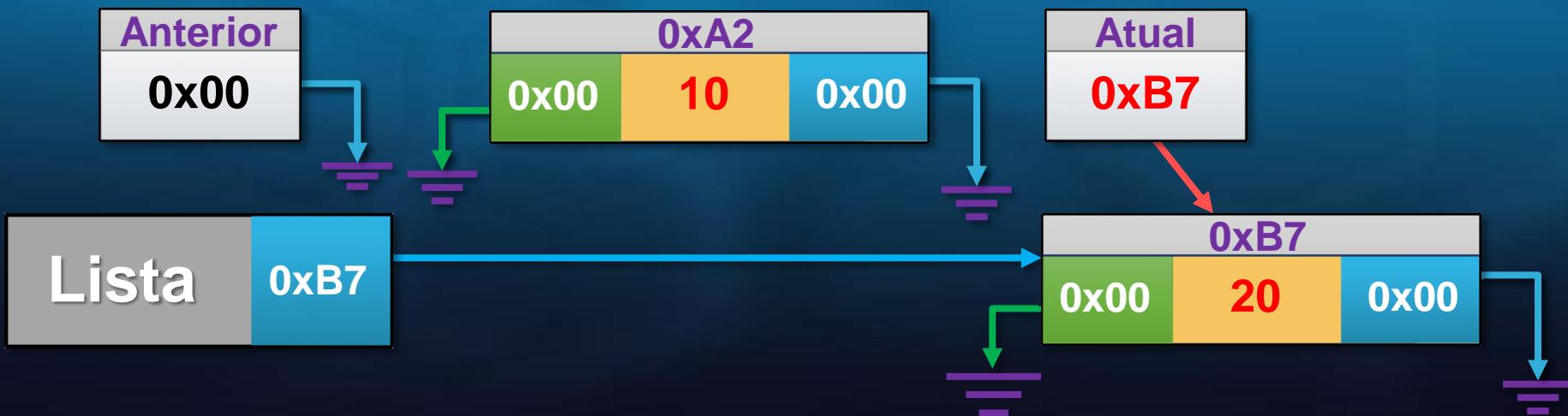
```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

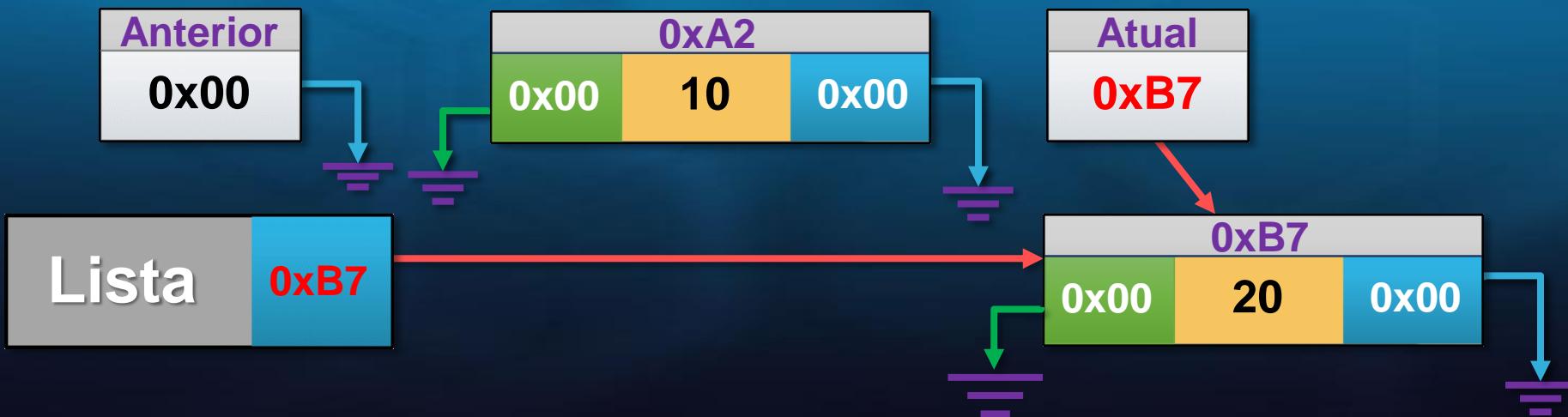
    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

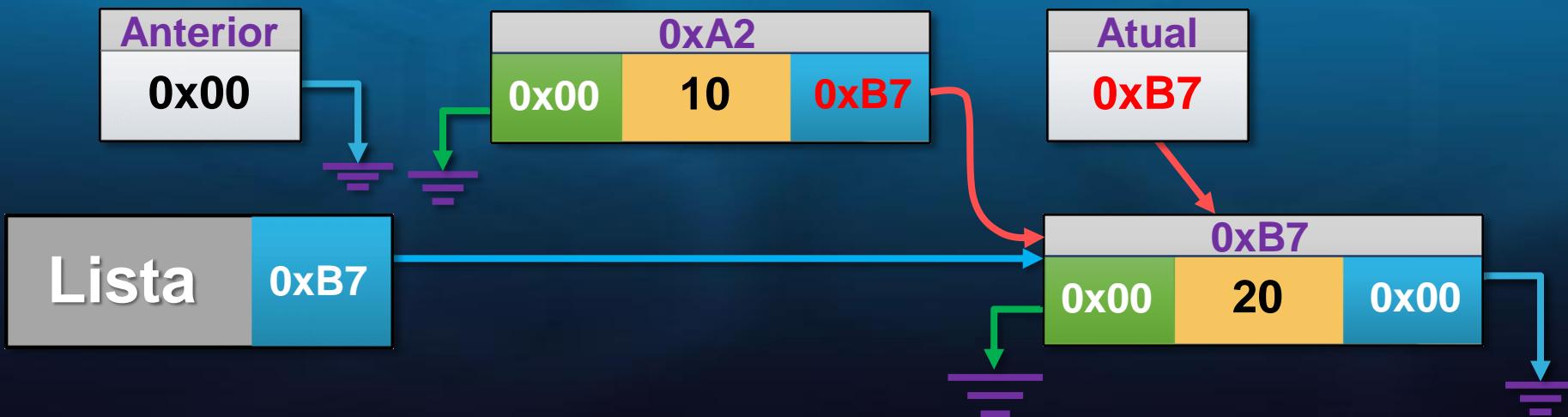
20 < 10



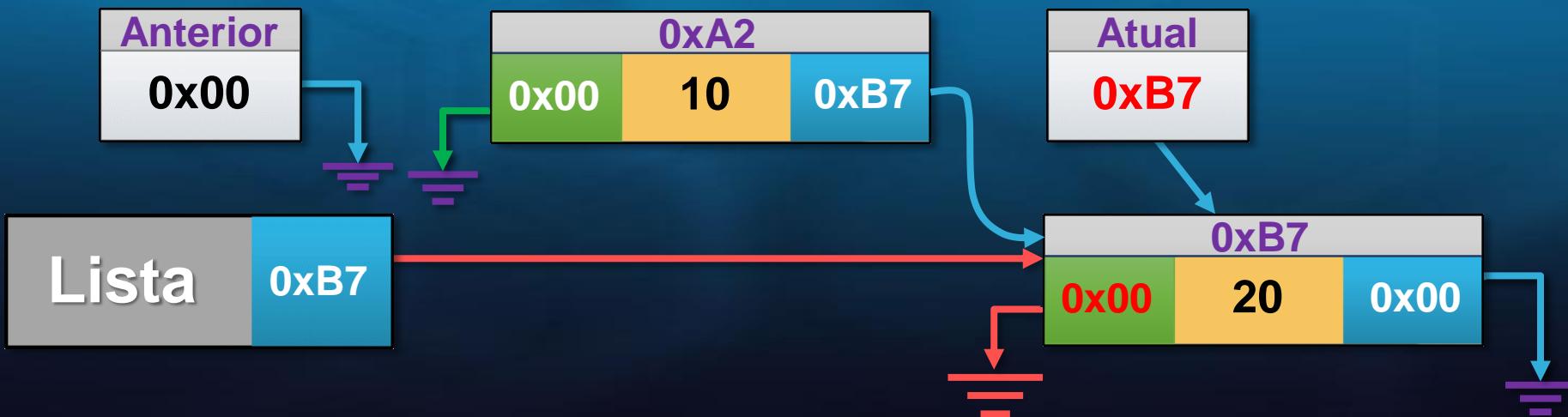
```
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->antNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
```



```
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->antNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
```



```
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->antNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
```



```
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->antNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
```



```
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->antNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
```

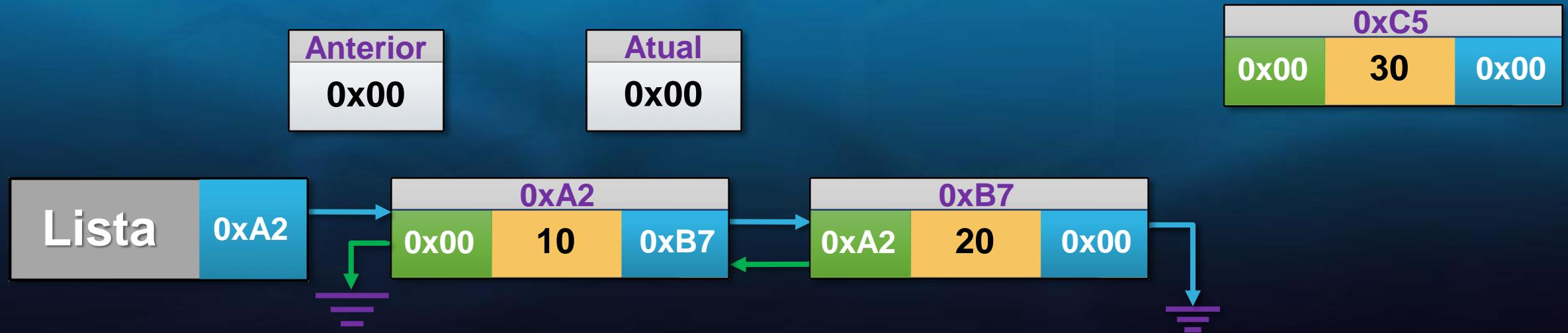


```
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->antNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
```

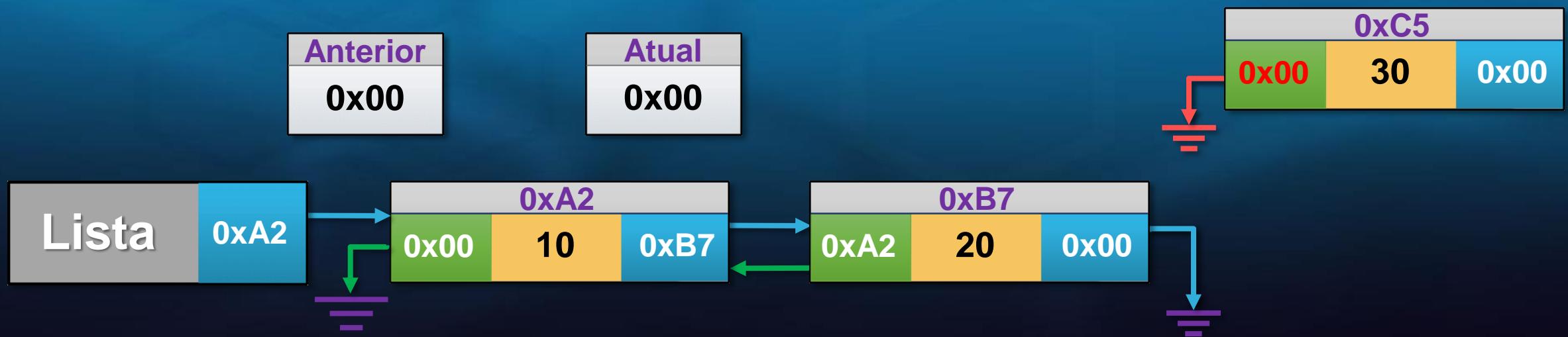


INSERIR ORDENADO NA LISTA

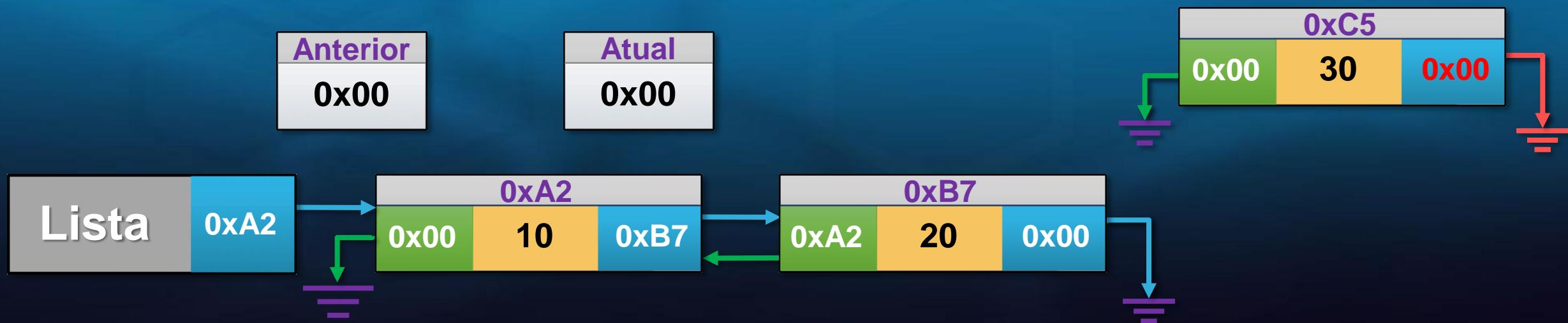
SIMULAÇÃO 3



```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;
```



```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;
```

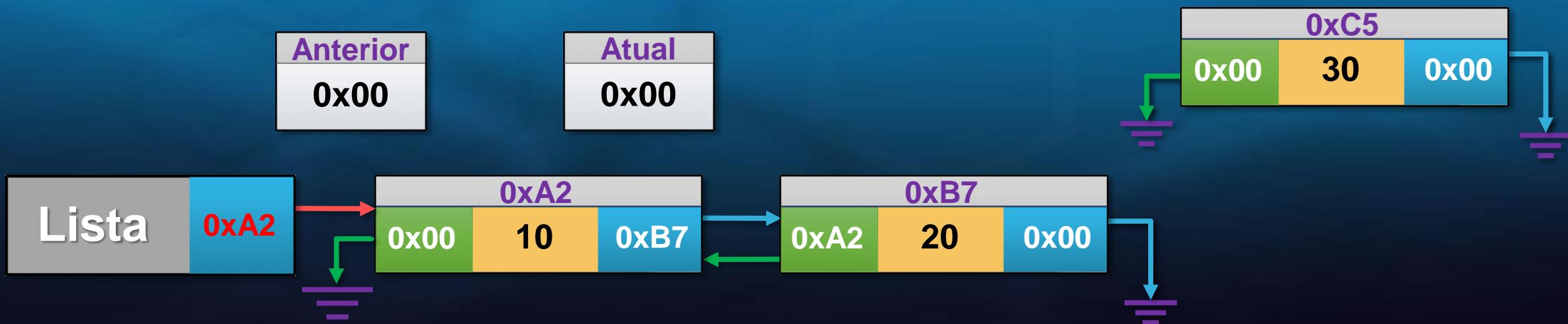


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

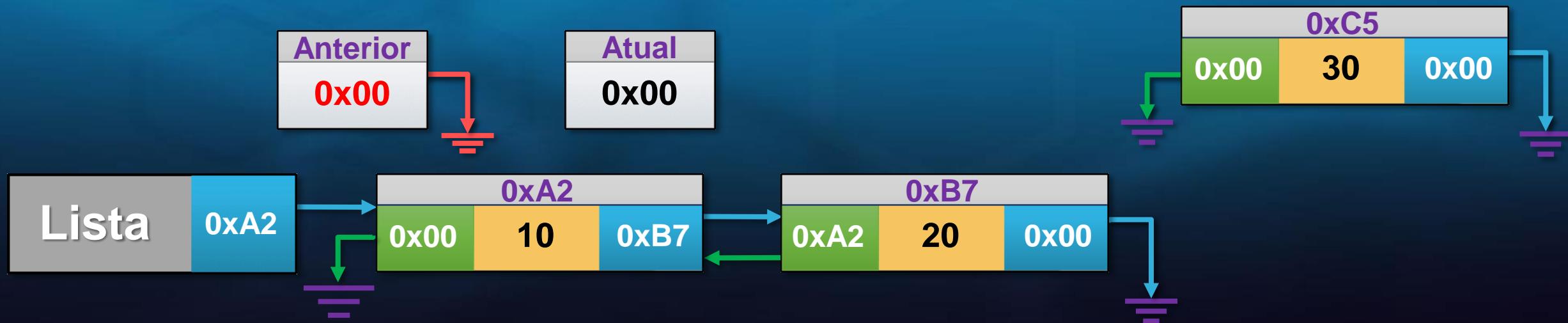


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```



```

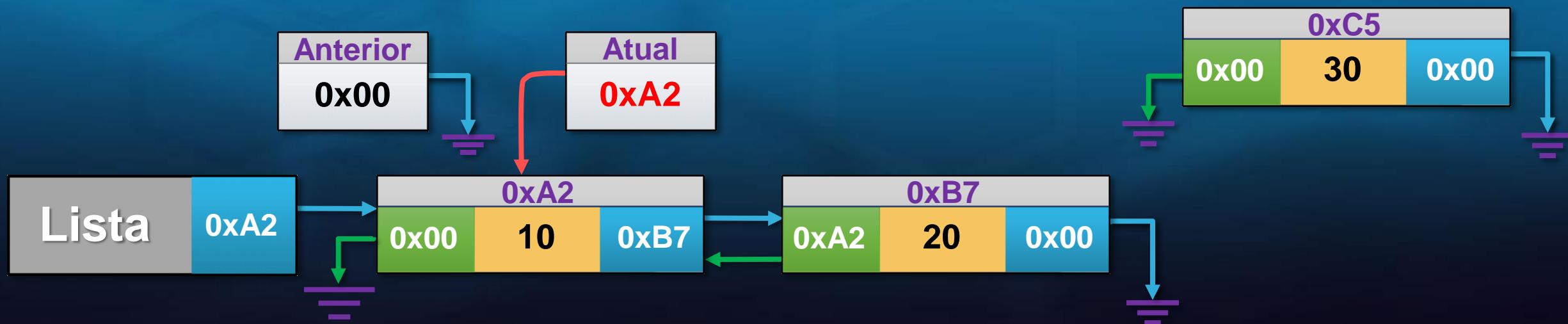
// Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;
}

```

```

// Localiza a posição de inserção
while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

```



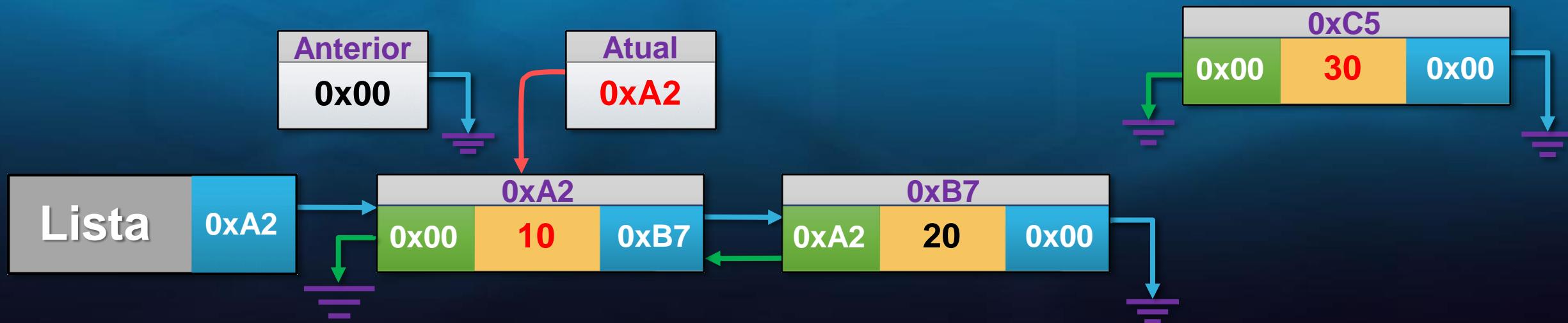
```

// Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

10 < 30

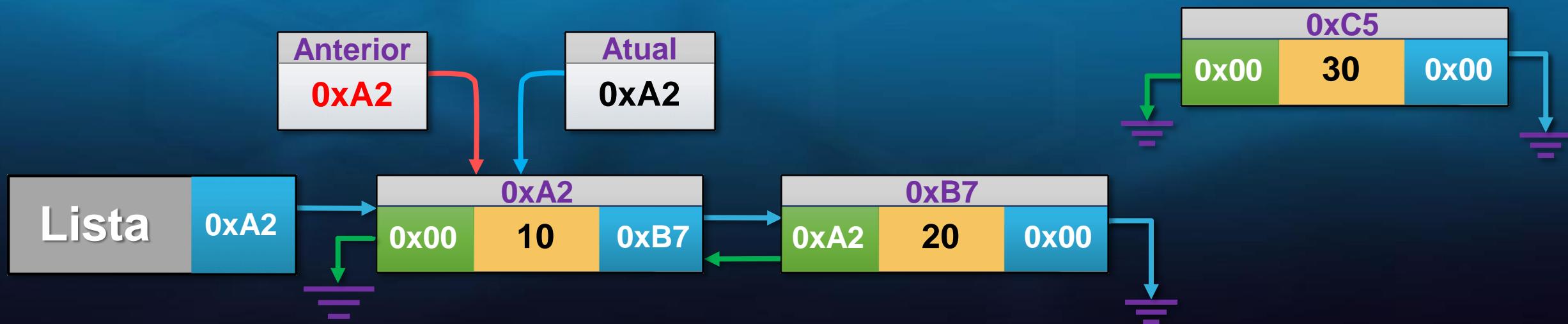


```

// Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

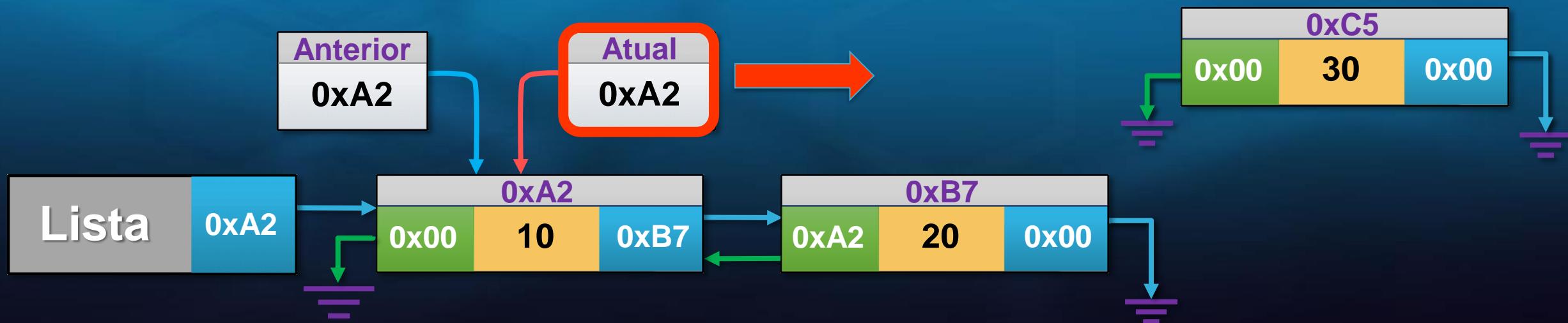


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

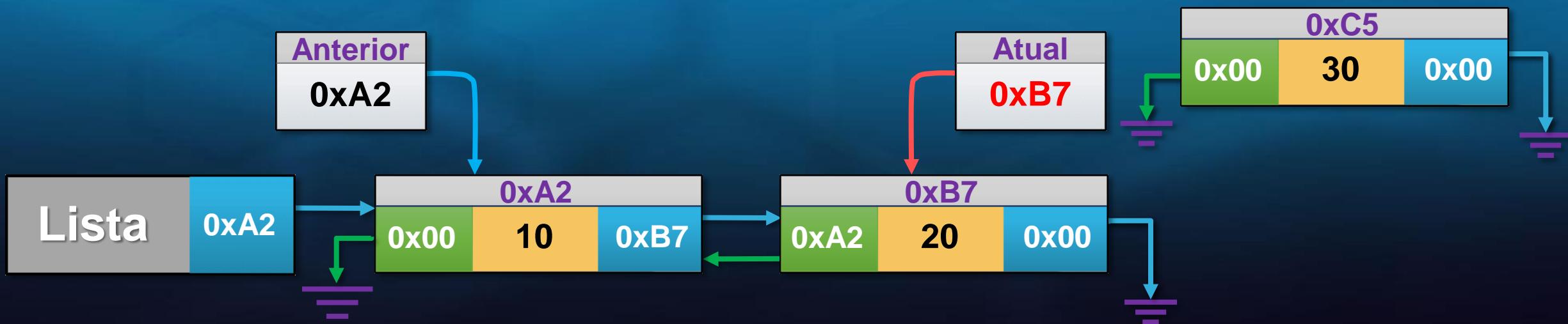


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```



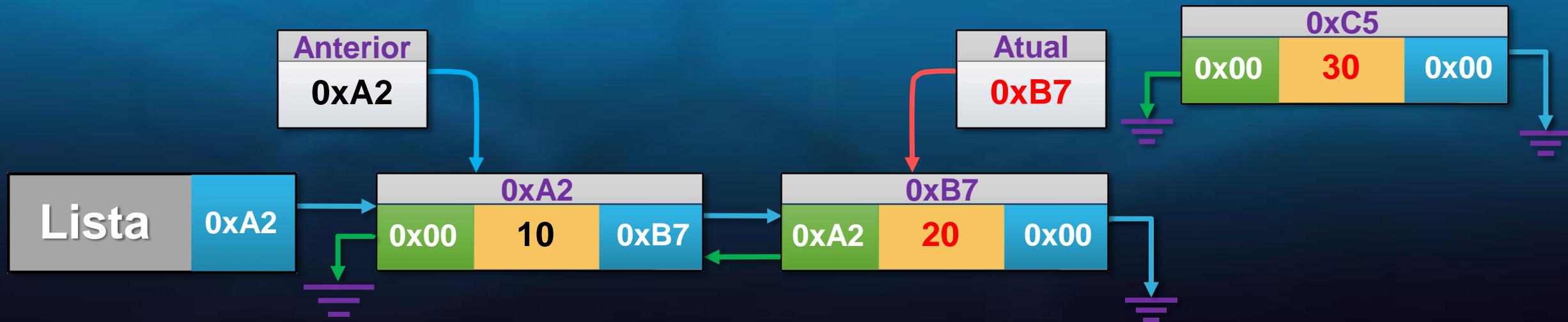
```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

20 < 30

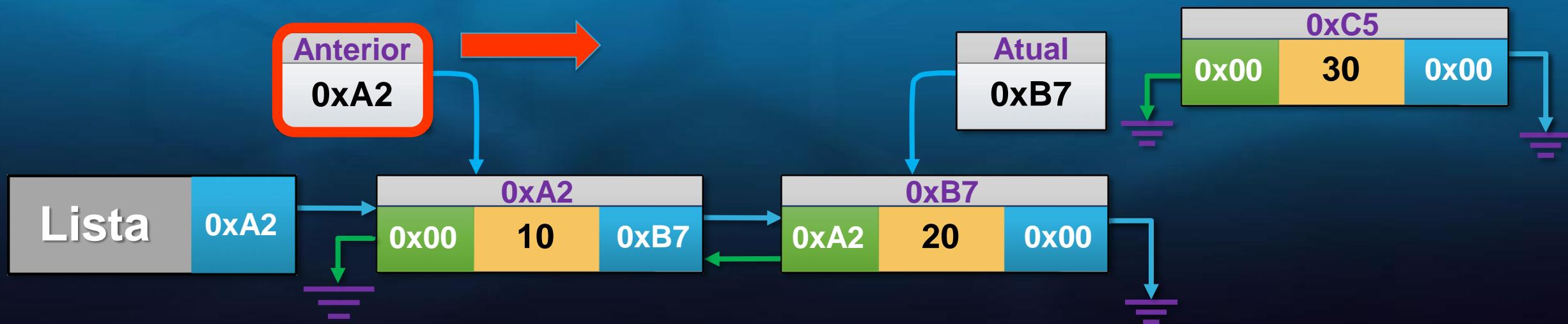


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

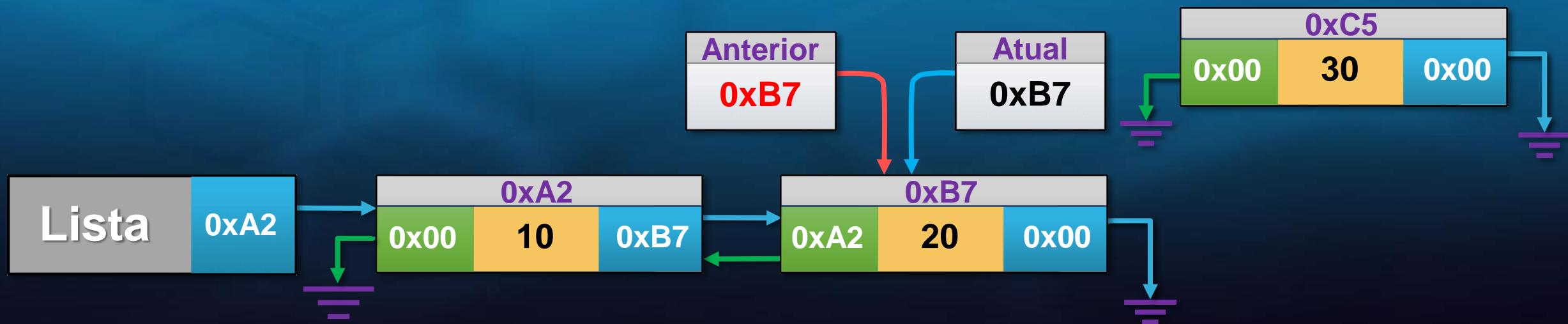


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

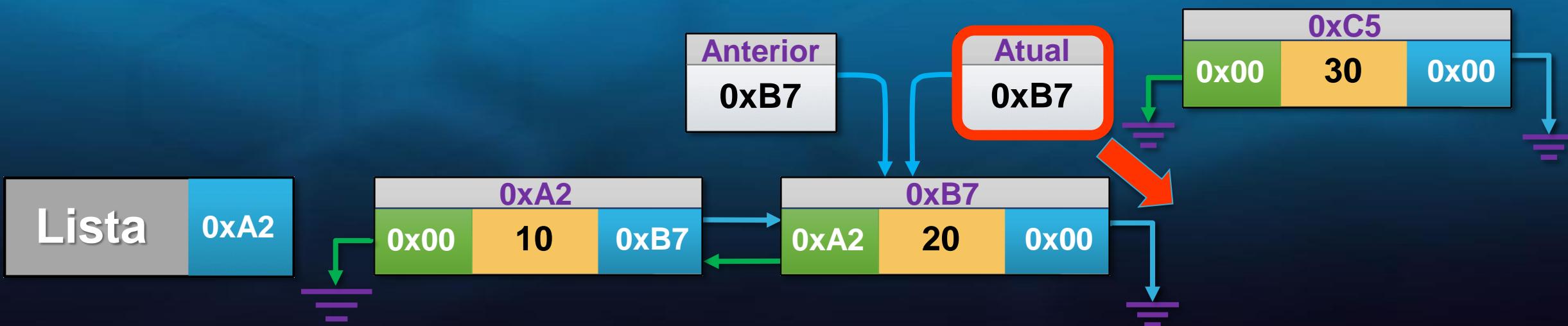


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

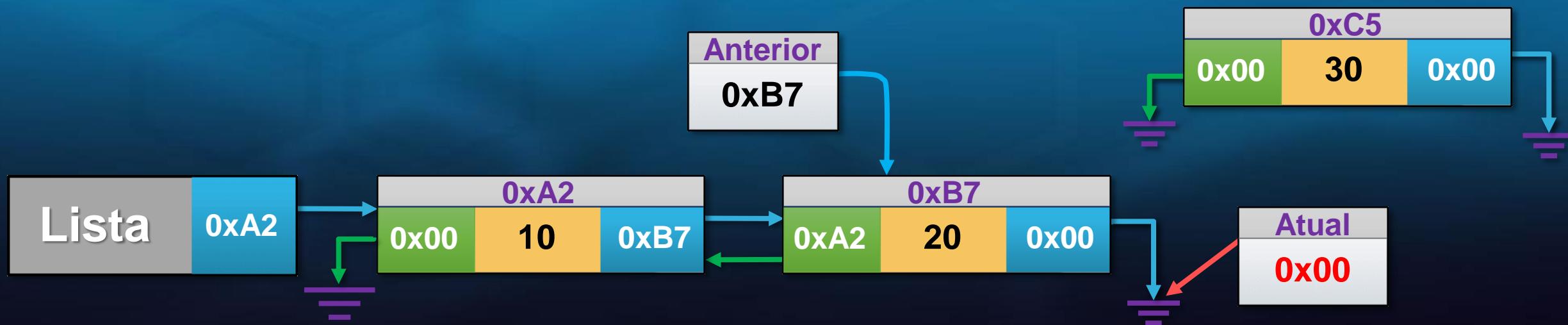


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```



```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;
}

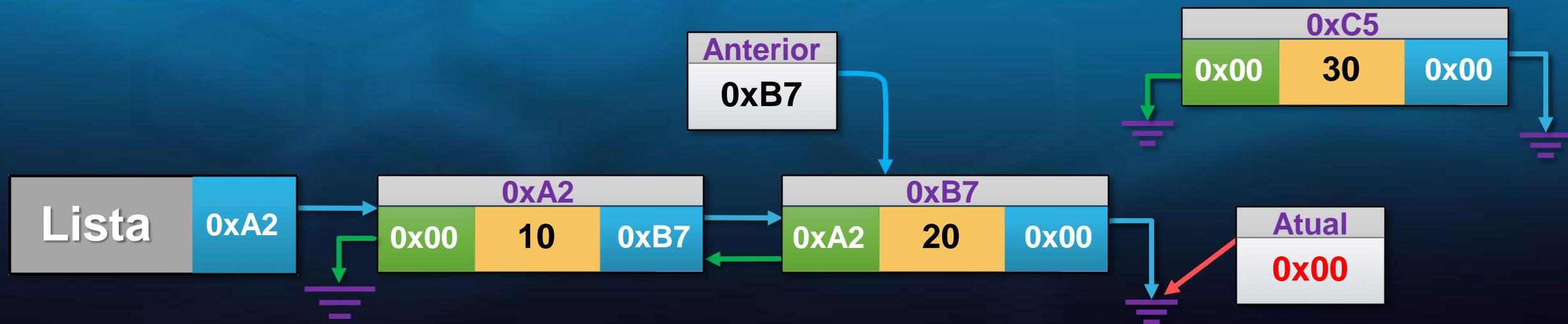
```

FALSO

```

while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

```



```

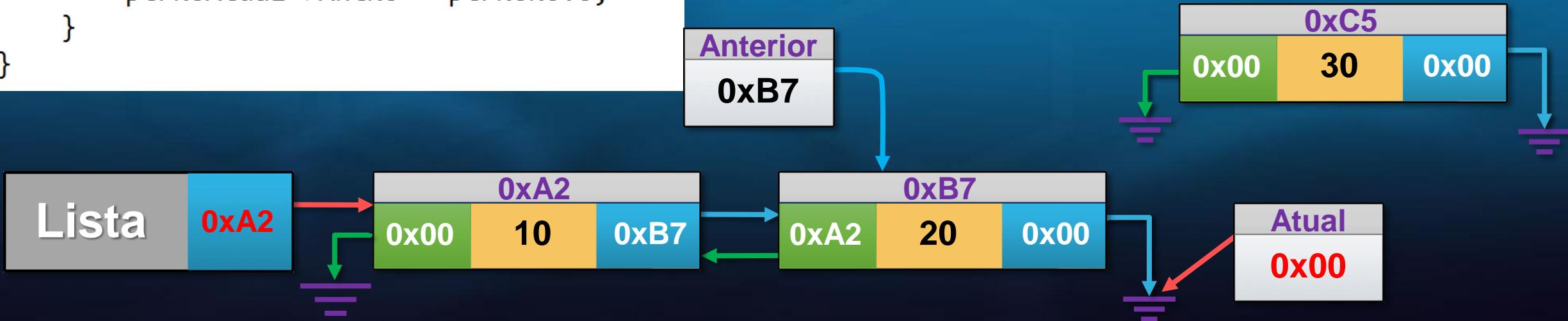
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}

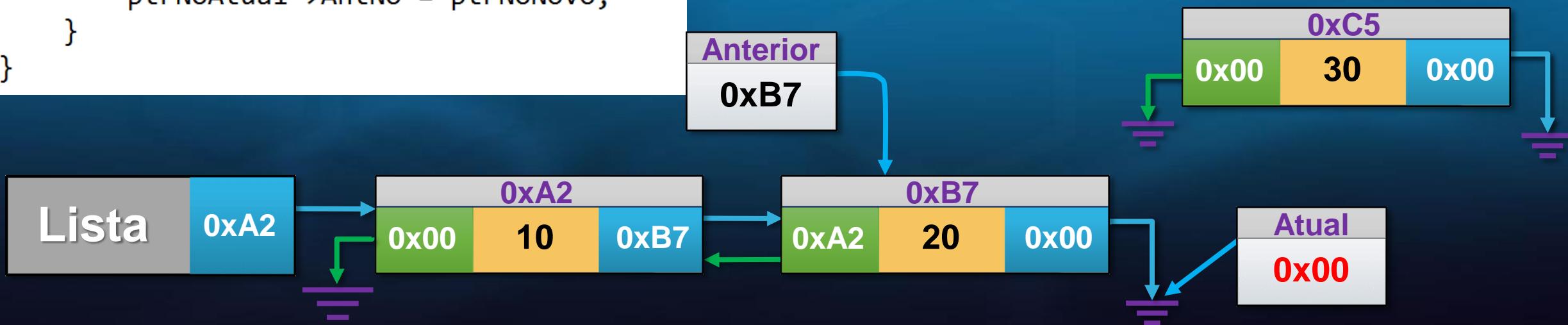
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

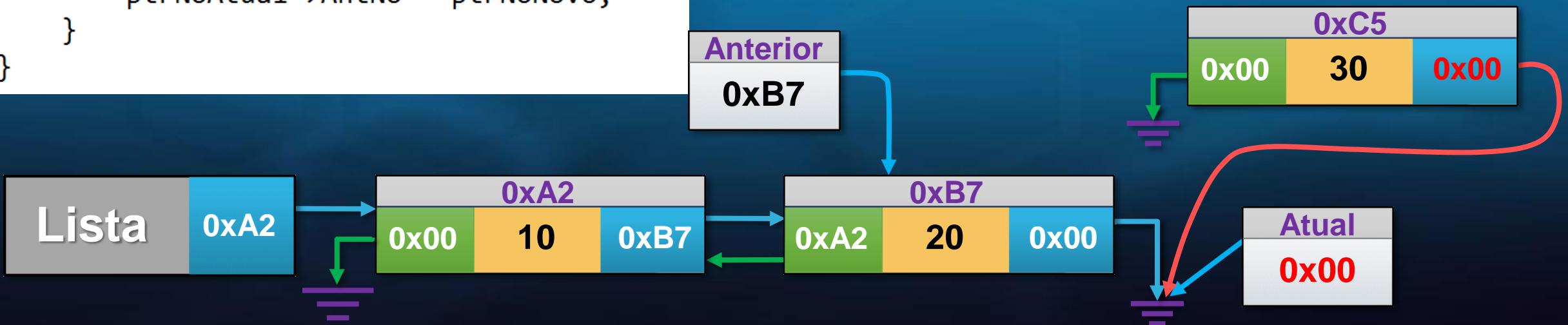
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

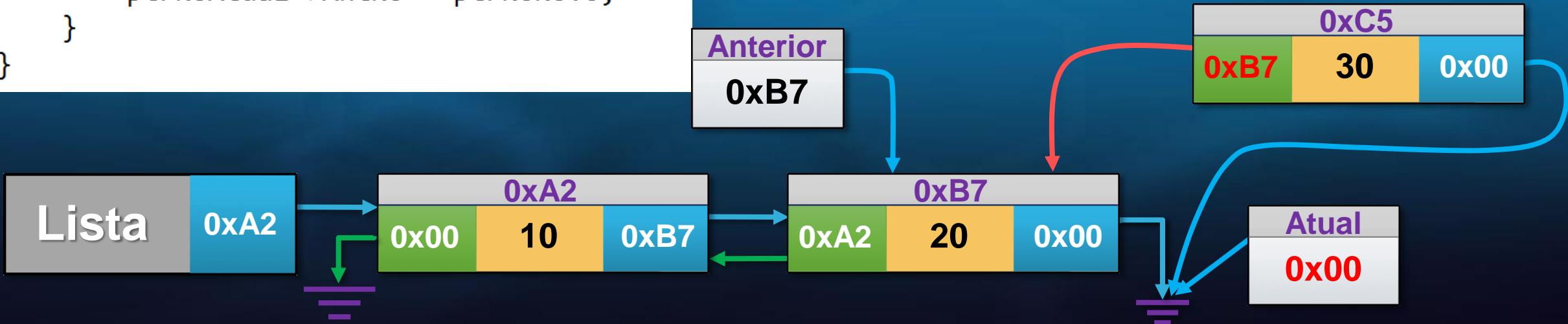
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

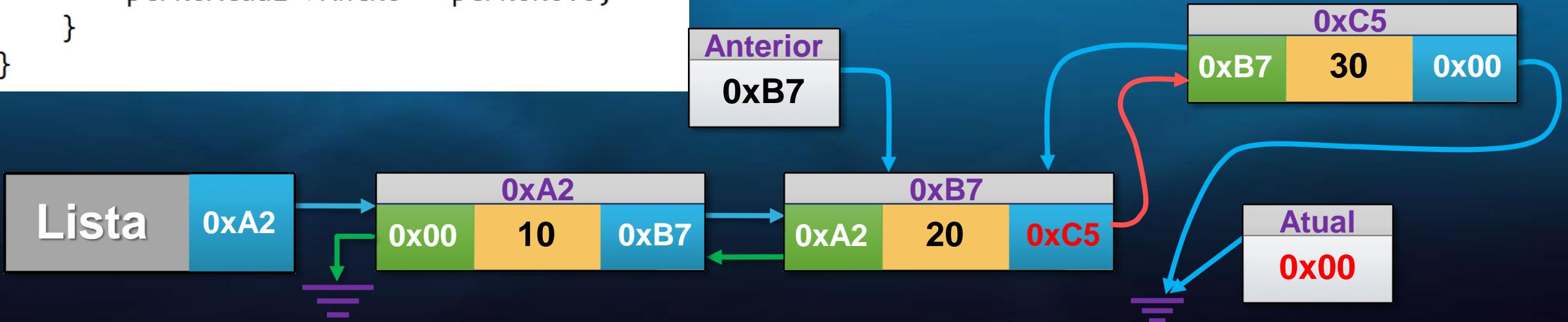
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

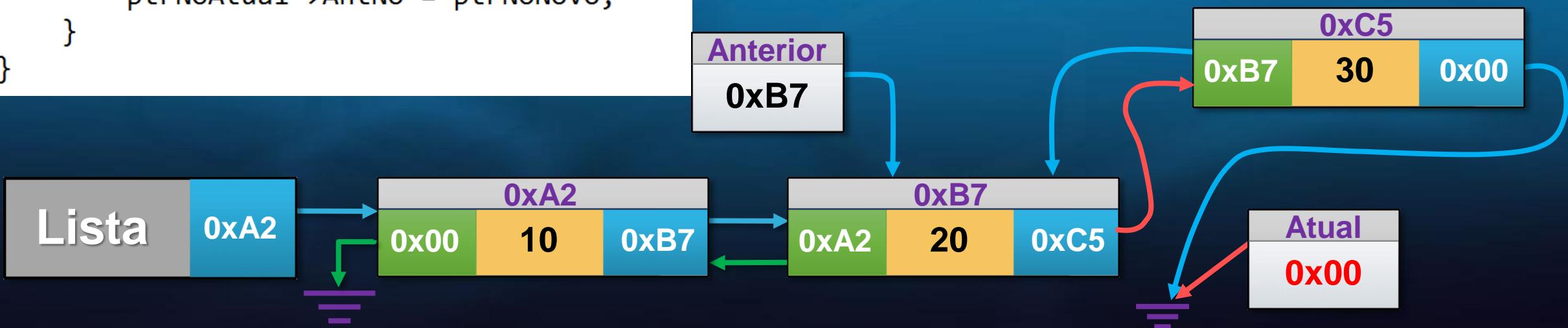
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

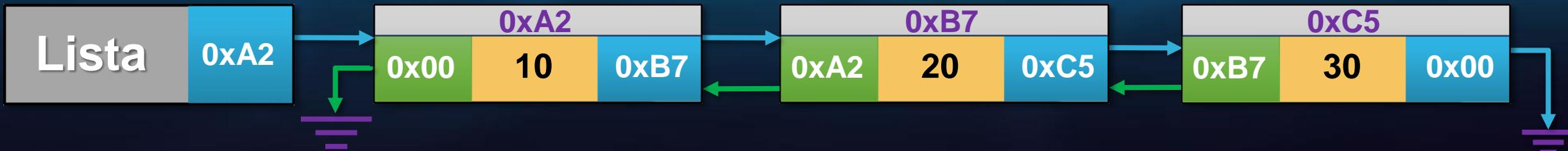
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

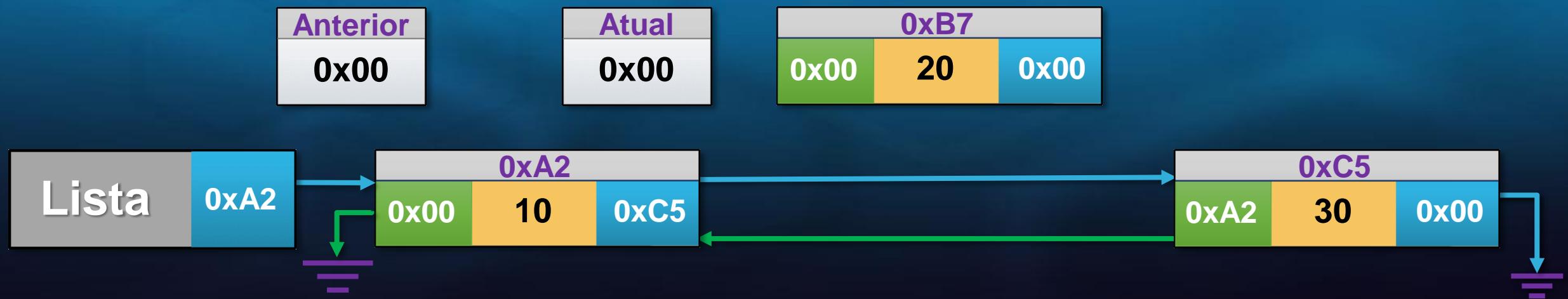
    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```

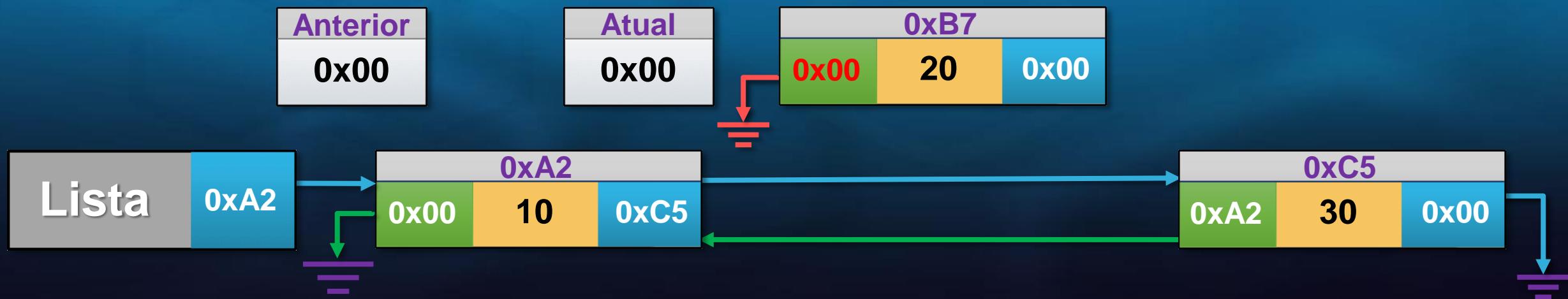


INSERIR ORDENADO NA LISTA

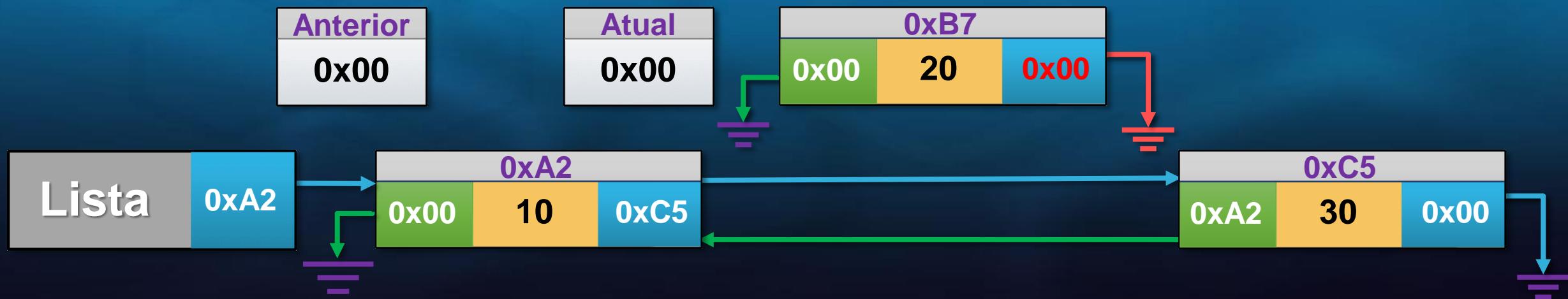
SIMULAÇÃO 4



```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;
```



```
ptrNoNovo->AntNo = NULL;  
ptrNoNovo->proxNo = NULL;
```

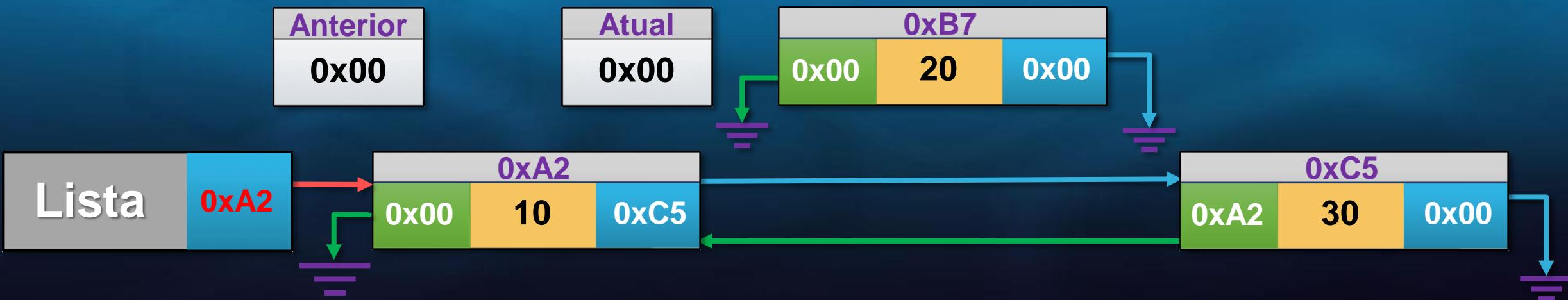


```

// Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

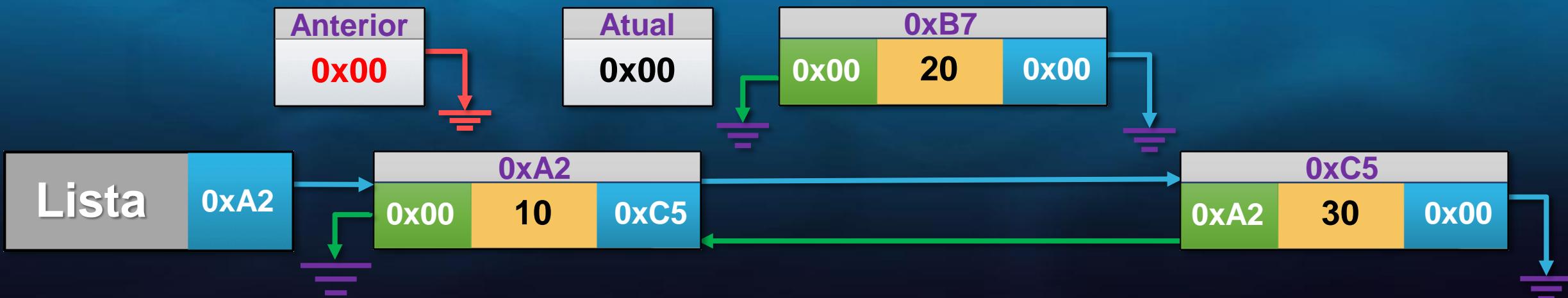


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

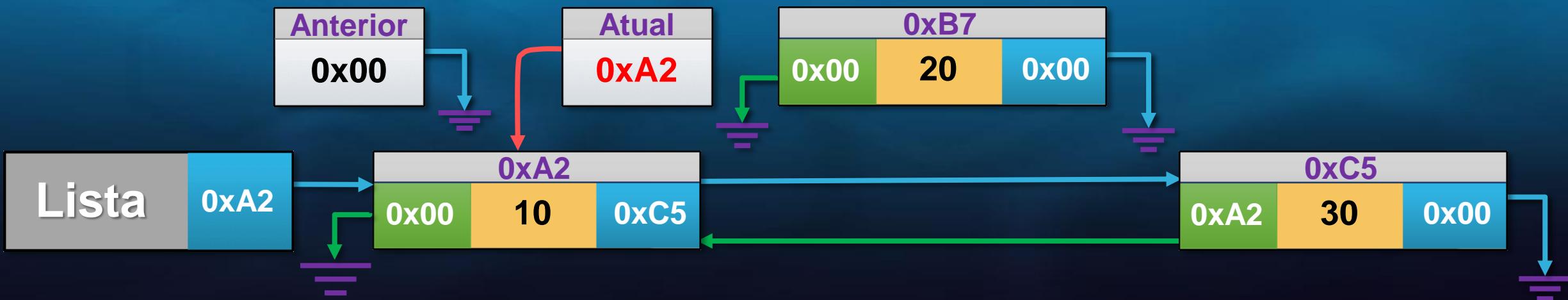


```

// Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```



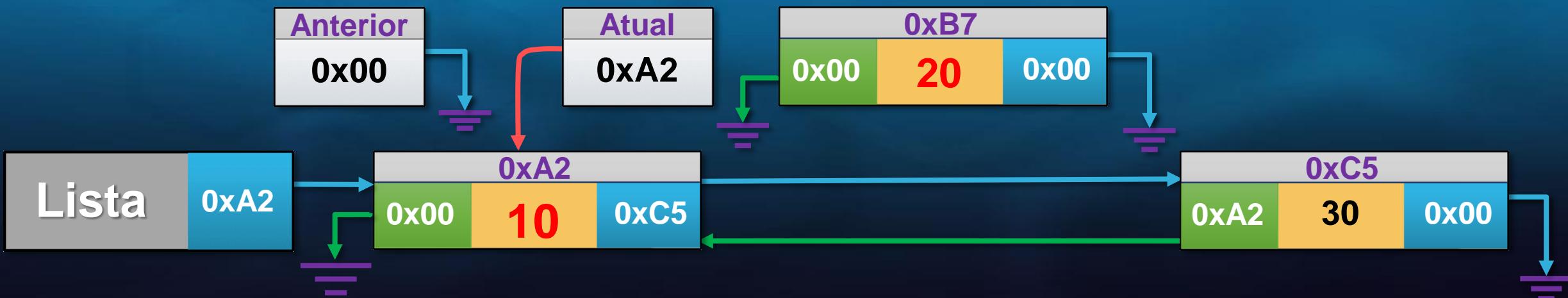
```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

10 < 20

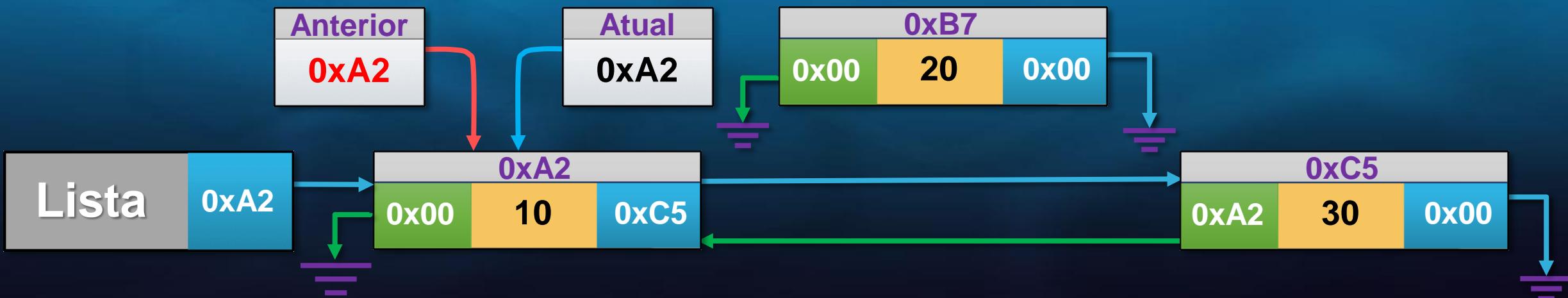


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

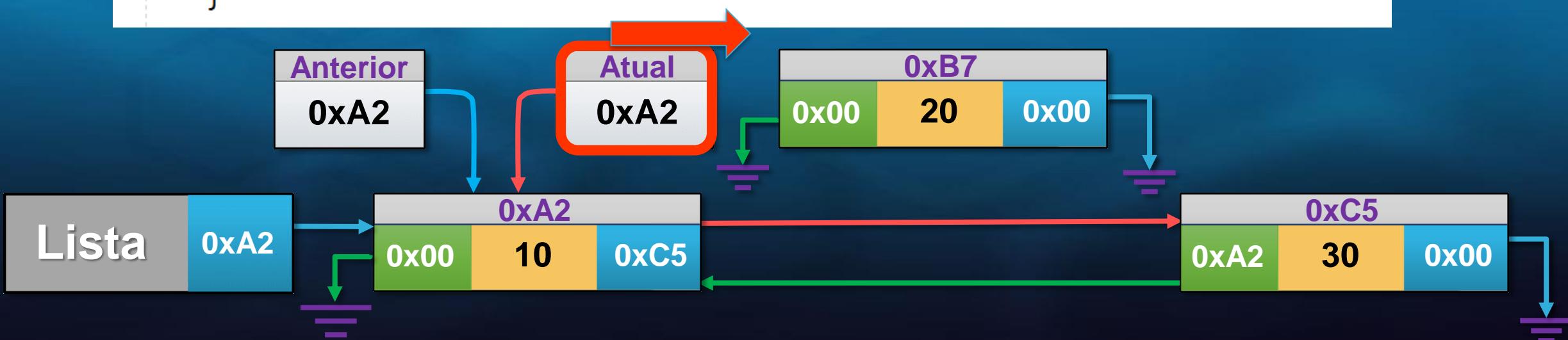


```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

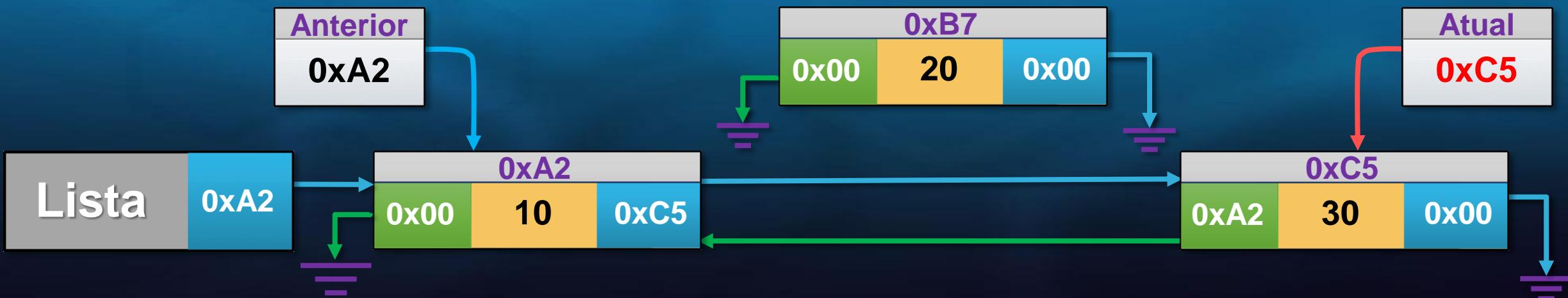


```

// Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```



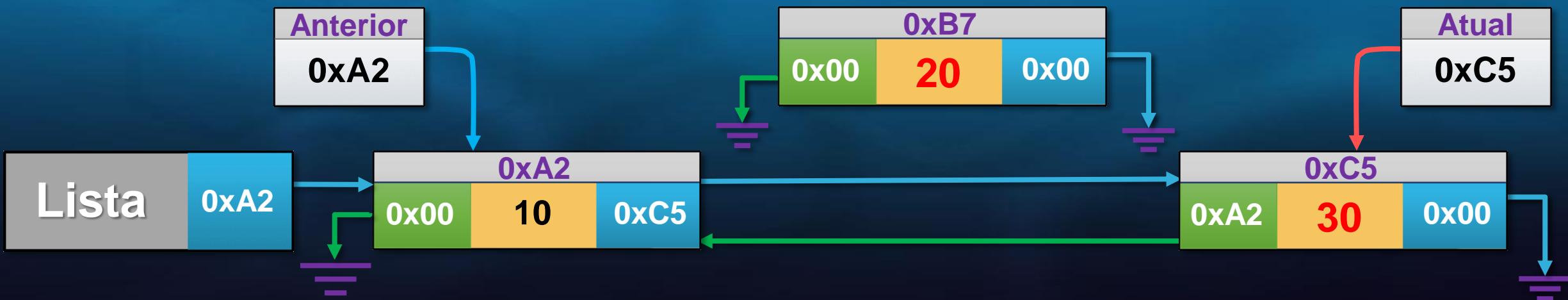
```

//Se a lista estiver vazia
if (ptrLista->inicio == NULL) {
    ptrLista->inicio = ptrNoNovo;
}
else
{
    ptrNoAnterior = NULL;
    ptrNoAtual = ptrLista->inicio;

    // Localiza a posição de inserção
    while (ptrNoAtual != NULL && ptrNoAtual->dados.matricula < matricula) {
        ptrNoAnterior = ptrNoAtual;
        ptrNoAtual = ptrNoAtual->proxNo;
    }
}

```

30 < 20



```

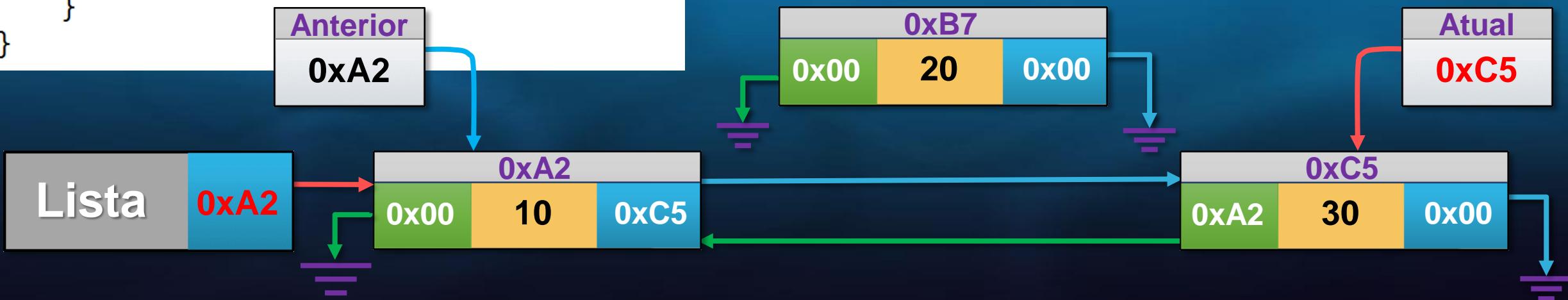
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}

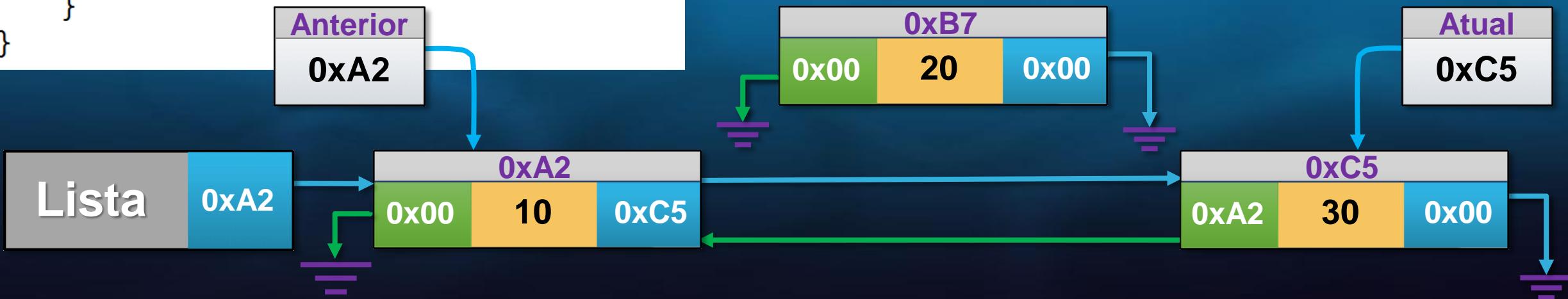
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

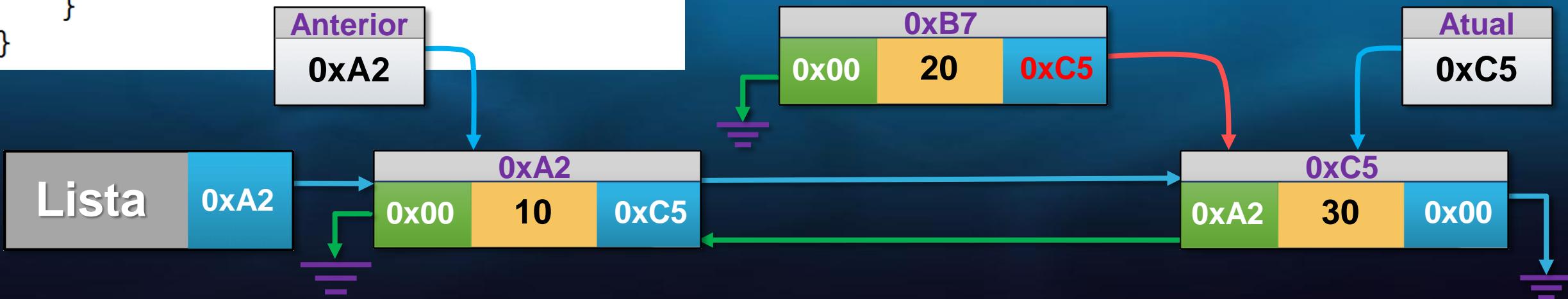
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

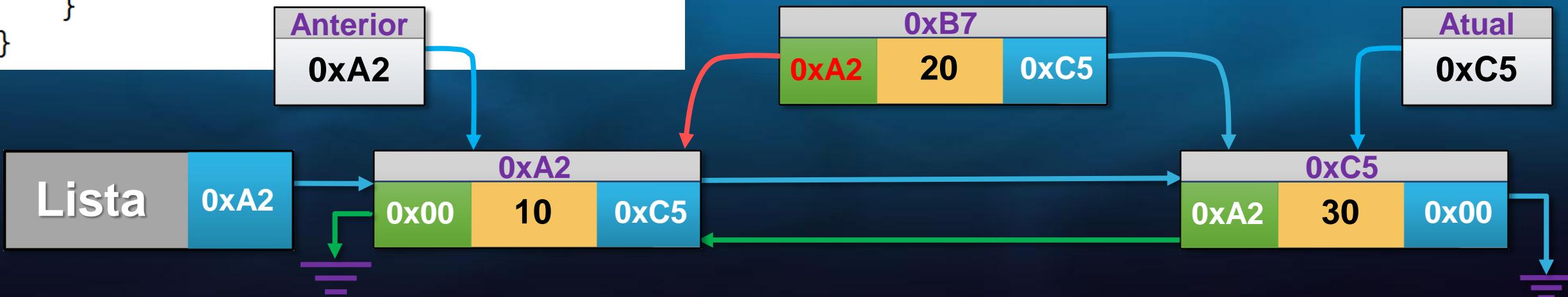
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

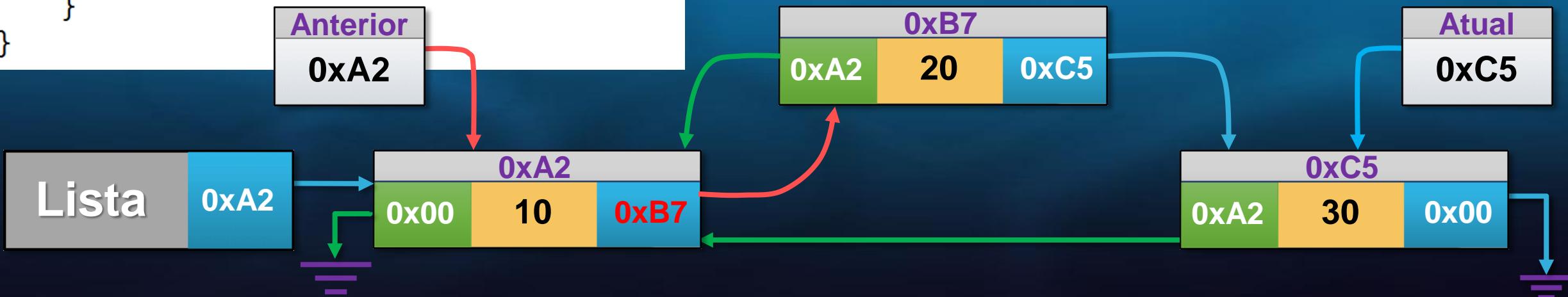
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

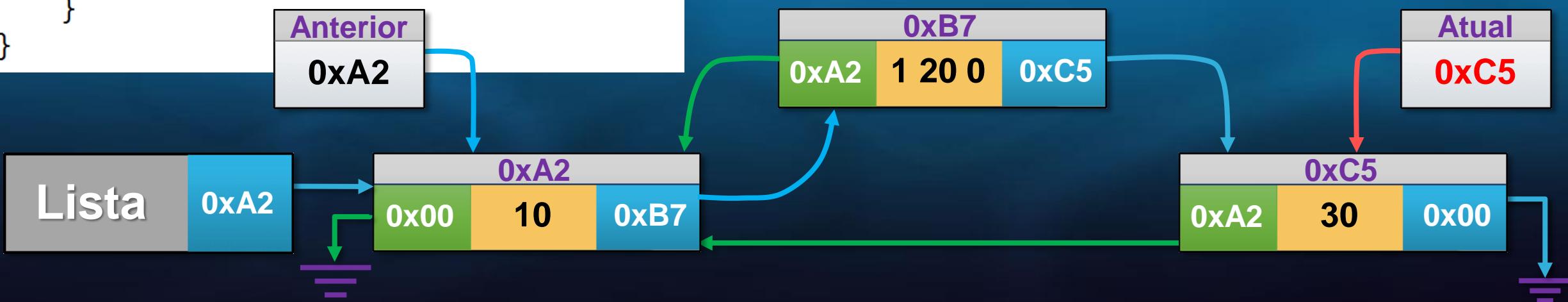
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

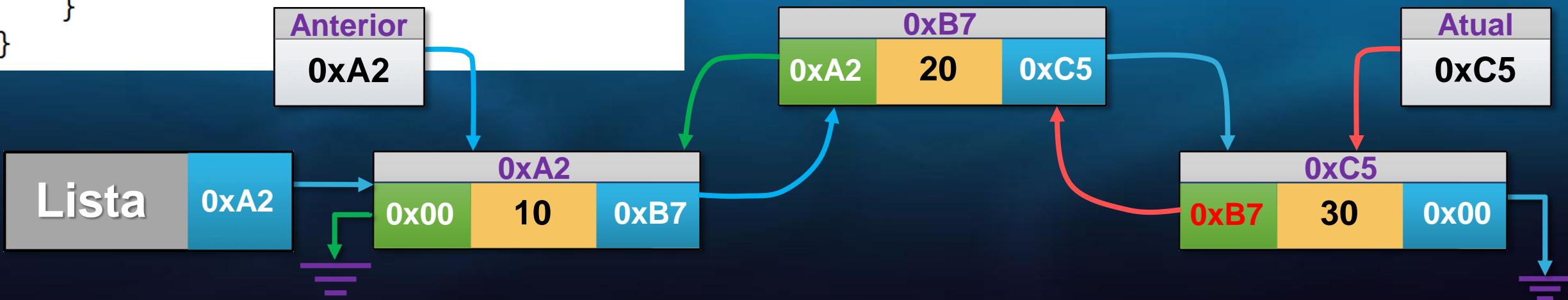
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



```

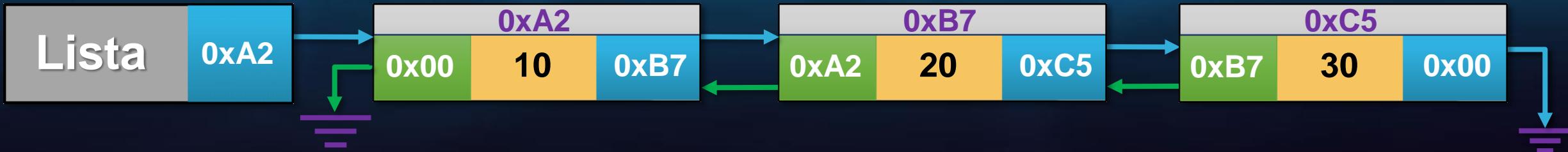
// Insere no INÍCIO da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrNoNovo->proxNo = ptrLista->inicio;
    ptrLista->inicio->AntNo = ptrNoNovo;
    ptrLista->inicio = ptrNoNovo;
}
else { // Insere no MEIO ou FIM da lista

    ptrNoNovo->proxNo = ptrNoAtual;
    ptrNoNovo->AntNo = ptrNoAnterior;

    ptrNoAnterior->proxNo = ptrNoNovo;

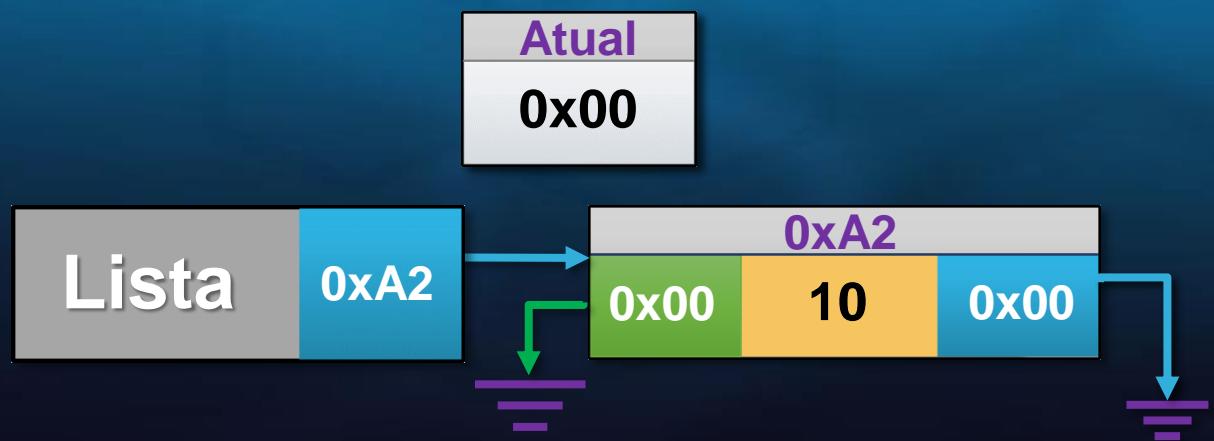
    // Se for o meio da lista
    if (ptrNoAtual != NULL) {
        ptrNoAtual->AntNo = ptrNoNovo;
    }
}

```



EXCLUIR NO INÍCIO DA LISTA

SIMULAÇÃO 1

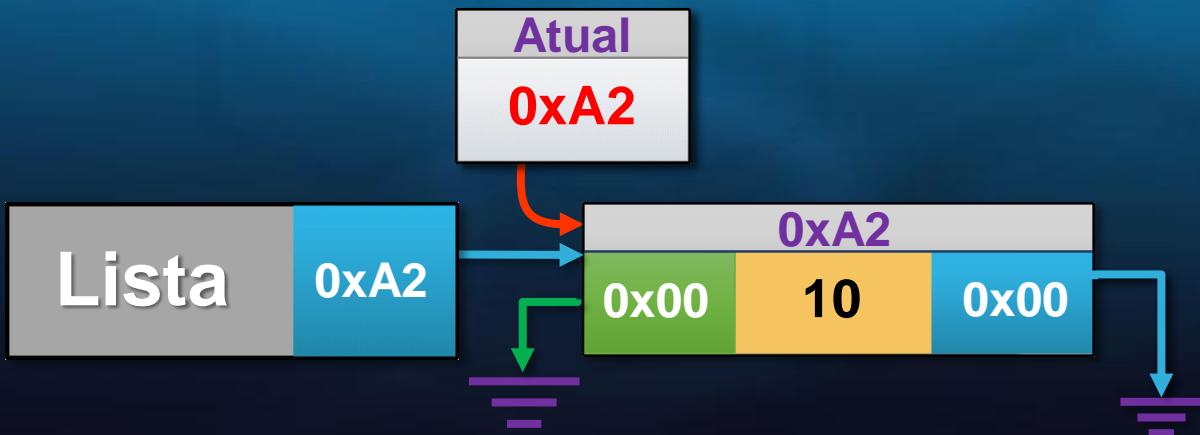


```
// Ajusta o INÍCIO
ptrNoAtual = ptrLista->inicio;

ptrLista->inicio = ptrNoAtual->proxNo;

// Se houver mais que um nó na lista
if (ptrNoAtual->proxNo != NULL) {
    ptrNoAtual->proxNo->antNo = NULL;
}

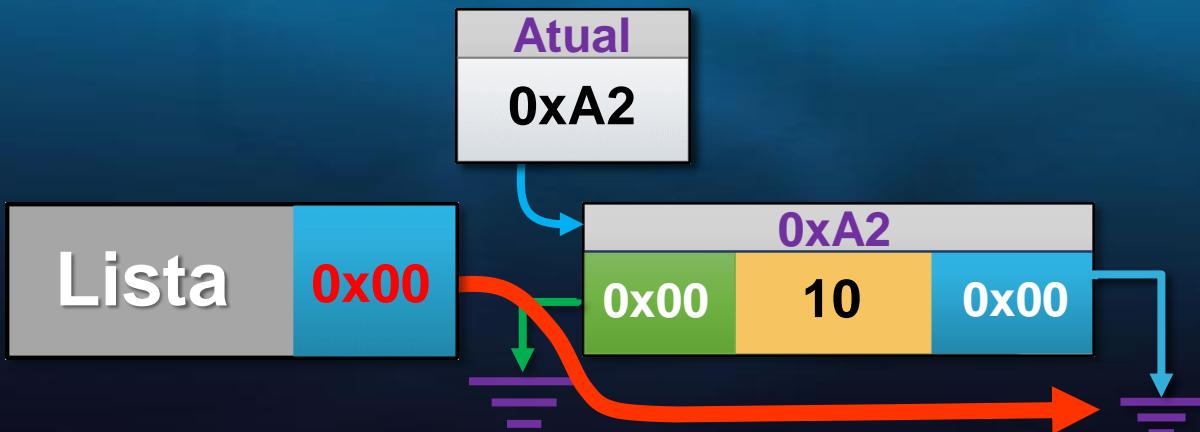
// Exclui o primeiro nó
delete ptrNoAtual;
```



```
// Ajusta o INÍCIO  
ptrNoAtual = ptrLista->inicio;  
  
ptrLista->inicio = ptrNoAtual->proxNo;
```

```
// Se houver mais que um nó na lista  
if (ptrNoAtual->proxNo != NULL) {  
    ptrNoAtual->proxNo->antNo = NULL;  
}
```

```
// Exclui o primeiro nó  
delete ptrNoAtual;
```

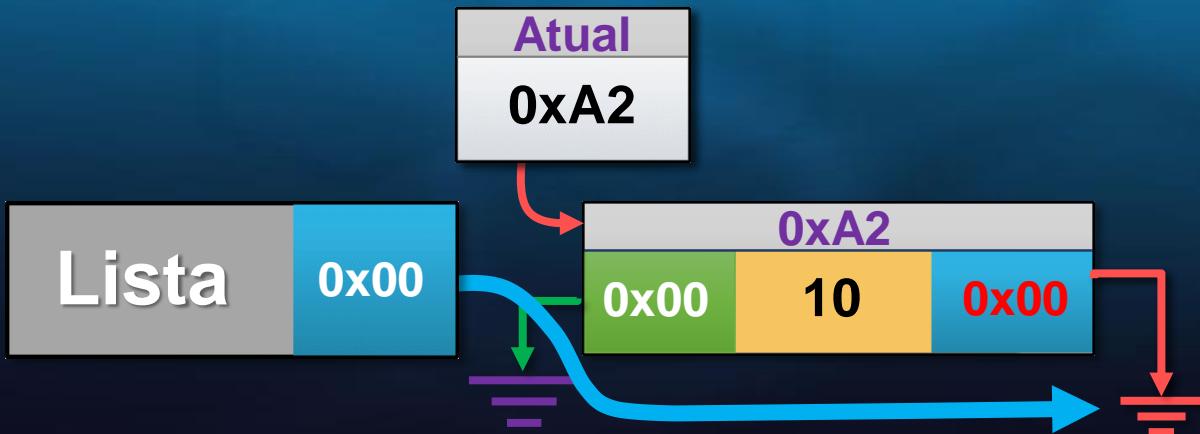


```
// Ajusta o INÍCIO
ptrNoAtual = ptrLista->inicio;

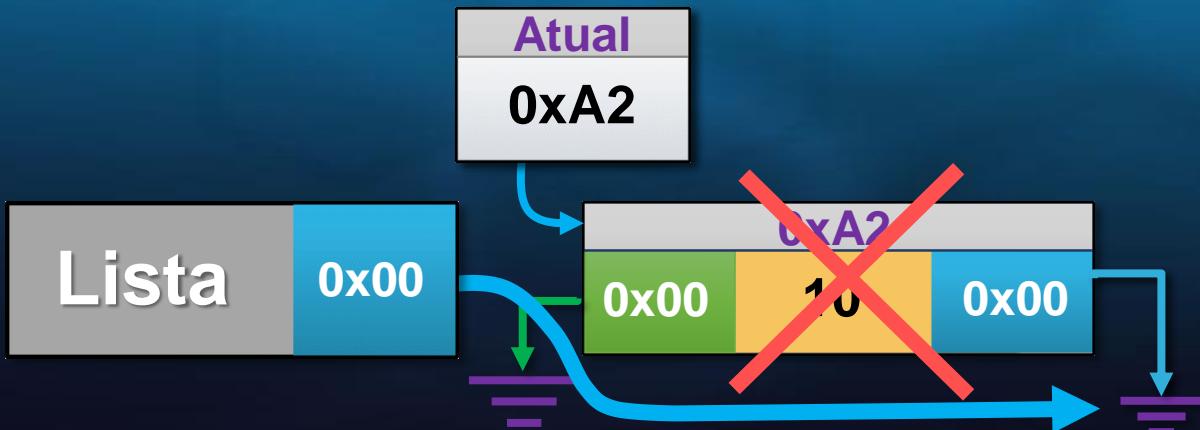
ptrLista->inicio = ptrNoAtual->proxNo;

// Se houver mais que um nó na lista
if (ptrNoAtual->proxNo != NULL) {
    ptrNoAtual->proxNo->antNo = NULL;
}

// Exclui o primeiro nó
delete ptrNoAtual;
```



```
// Ajusta o INÍCIO  
ptrNoAtual = ptrLista->inicio;  
  
ptrLista->inicio = ptrNoAtual->proxNo;  
  
// Se houver mais que um nó na lista  
if (ptrNoAtual->proxNo != NULL) {  
    ptrNoAtual->proxNo->antNo = NULL;  
}  
  
// Exclui o primeiro nó  
delete ptrNoAtual;
```



```
// Ajusta o INÍCIO
ptrNoAtual = ptrLista->inicio;

ptrLista->inicio = ptrNoAtual->proxNo;

// Se houver mais que um nó na lista
if (ptrNoAtual->proxNo != NULL) {
    ptrNoAtual->proxNo->antNo = NULL;
}

// Exclui o primeiro nó
delete ptrNoAtual;
```

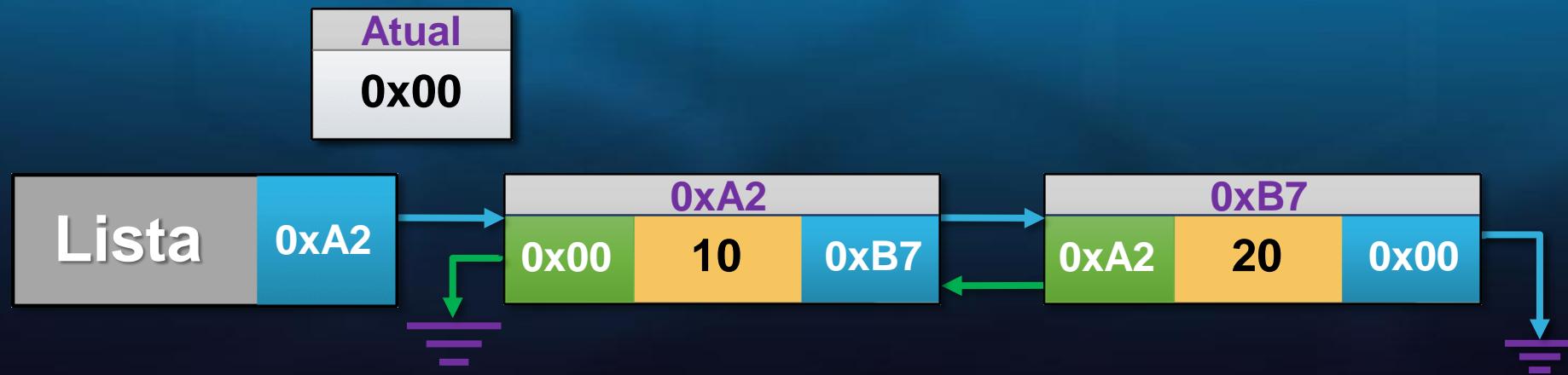
Lista

0x00



EXCLUIR NO INÍCIO DA LISTA

SIMULAÇÃO 2

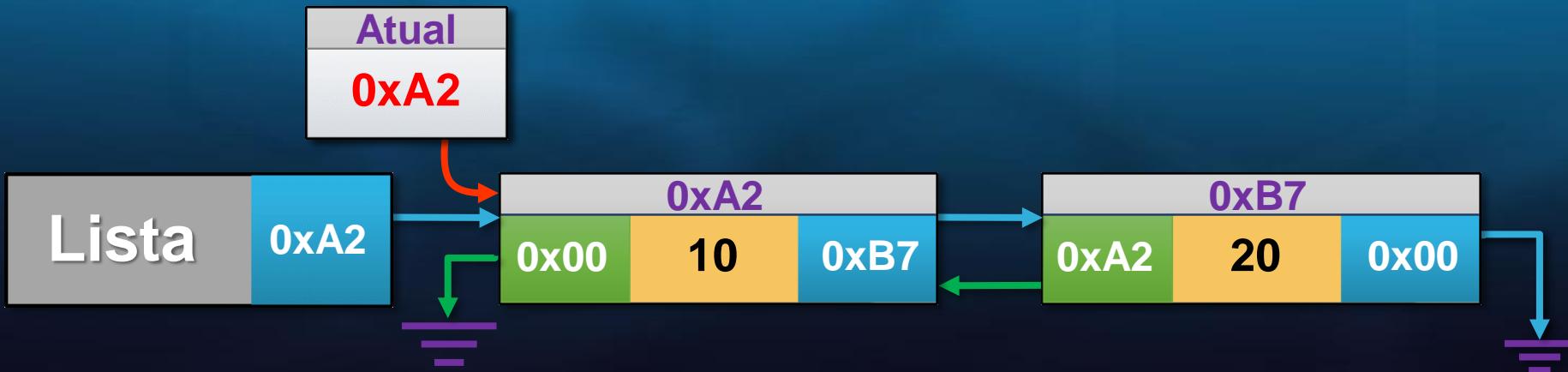


```
// Ajusta o INÍCIO
ptrNoAtual = ptrLista->inicio;

ptrLista->inicio = ptrNoAtual->proxNo;

// Se houver mais que um nó na lista
if (ptrNoAtual->proxNo != NULL) {
    ptrNoAtual->proxNo->antNo = NULL;
}

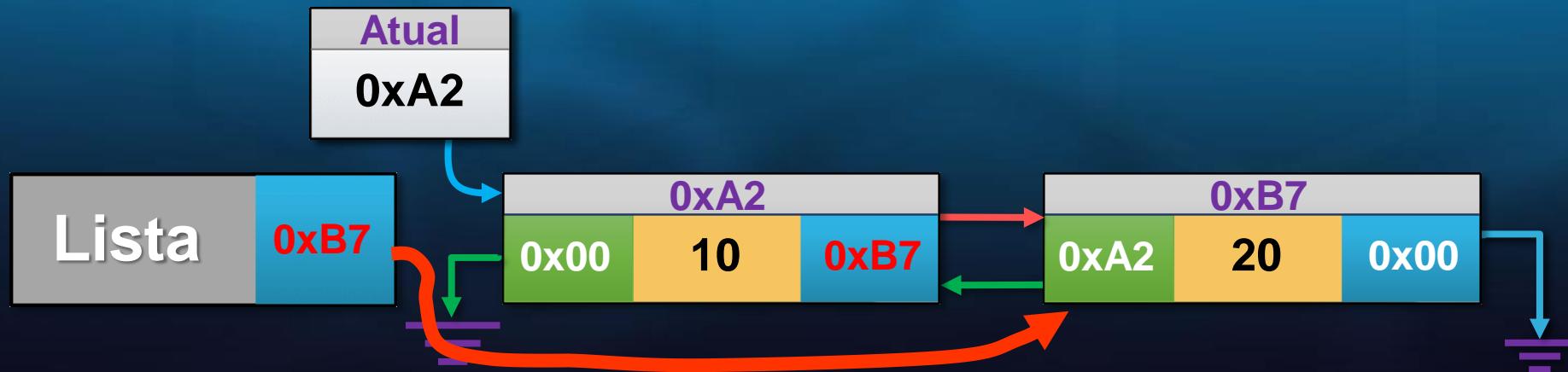
// Exclui o primeiro nó
delete ptrNoAtual;
```



```
// Ajusta o INÍCIO  
ptrNoAtual = ptrLista->inicio;  
  
ptrLista->inicio = ptrNoAtual->proxNo;
```

```
// Se houver mais que um nó na lista  
if (ptrNoAtual->proxNo != NULL) {  
    ptrNoAtual->proxNo->antNo = NULL;  
}
```

```
// Exclui o primeiro nó  
delete ptrNoAtual;
```

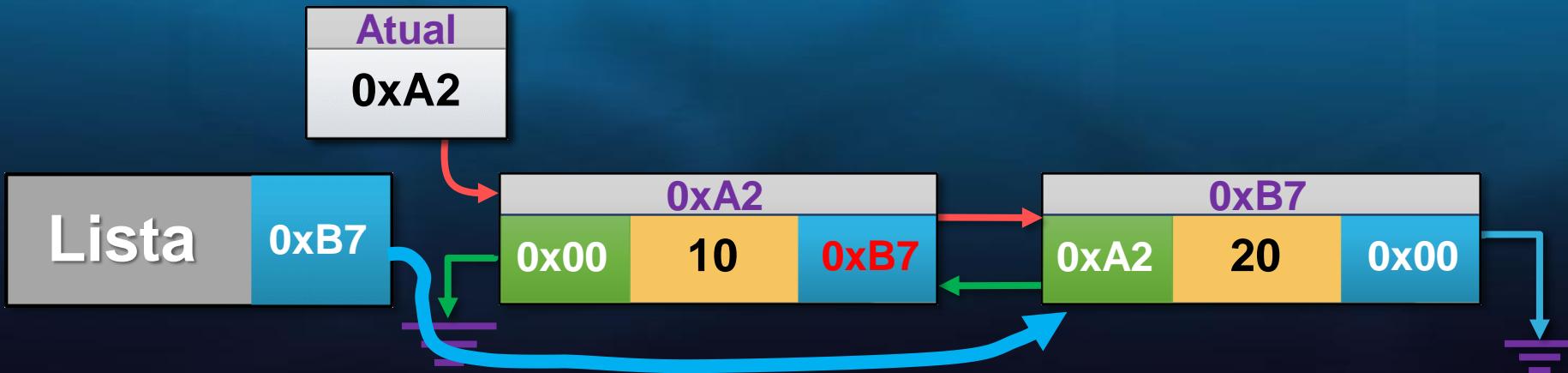


```
// Ajusta o INÍCIO
ptrNoAtual = ptrLista->inicio;

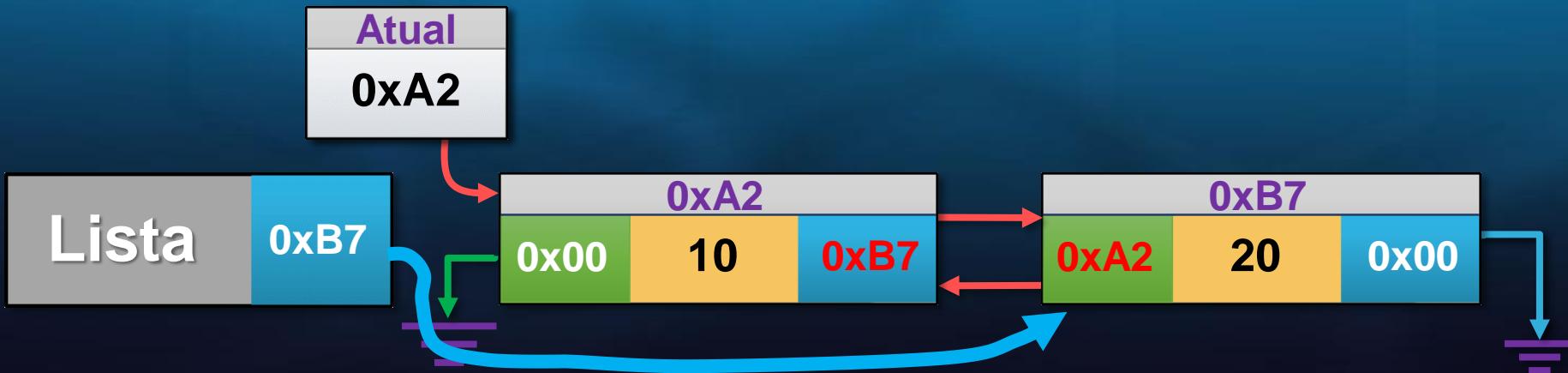
ptrLista->inicio = ptrNoAtual->proxNo;

// Se houver mais que um nó na lista
if (ptrNoAtual->proxNo != NULL) {
    ptrNoAtual->proxNo->antNo = NULL;
}

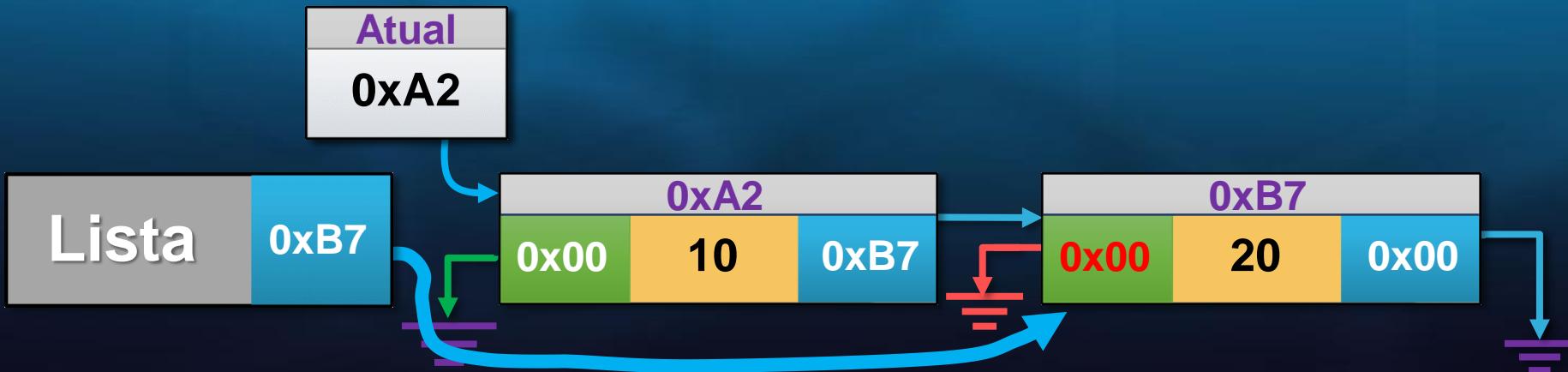
// Exclui o primeiro nó
delete ptrNoAtual;
```



```
// Ajusta o INÍCIO  
ptrNoAtual = ptrLista->inicio;  
  
ptrLista->inicio = ptrNoAtual->proxNo;  
  
// Se houver mais que um nó na lista  
if (ptrNoAtual->proxNo != NULL) {  
    ptrNoAtual->proxNo->antNo = NULL;  
}  
  
// Exclui o primeiro nó  
delete ptrNoAtual;
```



```
// Ajusta o INÍCIO  
ptrNoAtual = ptrLista->inicio;  
  
ptrLista->inicio = ptrNoAtual->proxNo;  
  
// Se houver mais que um nó na lista  
if (ptrNoAtual->proxNo != NULL) {  
    ptrNoAtual->proxNo->antNo = NULL;  
}  
  
// Exclui o primeiro nó  
delete ptrNoAtual;
```

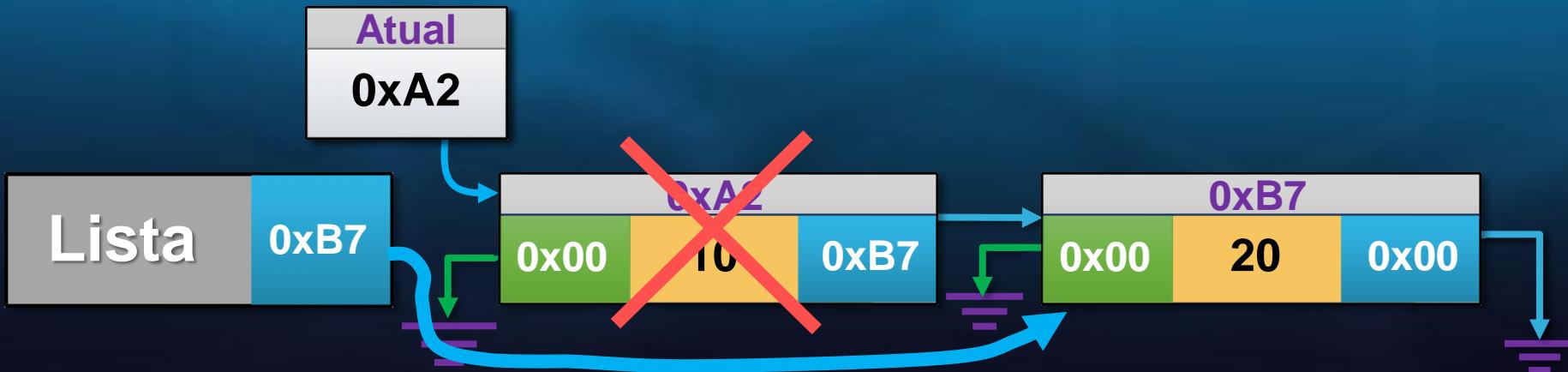


```
// Ajusta o INÍCIO
ptrNoAtual = ptrLista->inicio;

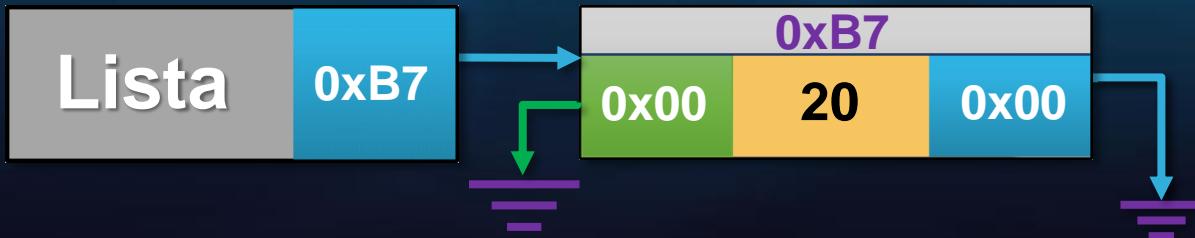
ptrLista->inicio = ptrNoAtual->proxNo;

// Se houver mais que um nó na lista
if (ptrNoAtual->proxNo != NULL) {
    ptrNoAtual->proxNo->antNo = NULL;
}

// Exclui o primeiro nó
delete ptrNoAtual;
```

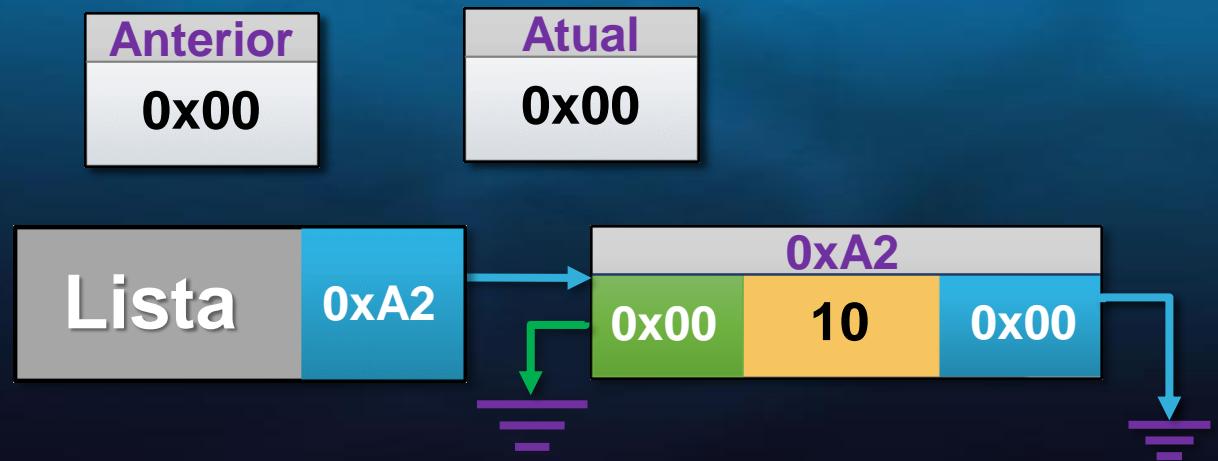


```
// Ajusta o INÍCIO  
ptrNoAtual = ptrLista->inicio;  
  
ptrLista->inicio = ptrNoAtual->proxNo;  
  
// Se houver mais que um nó na lista  
if (ptrNoAtual->proxNo != NULL) {  
    ptrNoAtual->proxNo->antNo = NULL;  
}  
  
// Exclui o primeiro nó  
delete ptrNoAtual;
```

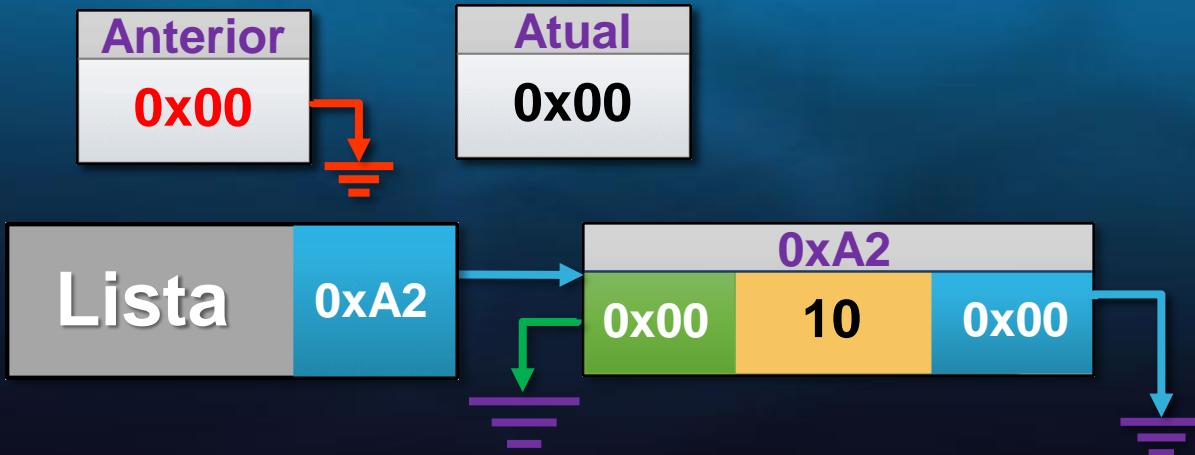


EXCLUIR NO FINAL DA LISTA

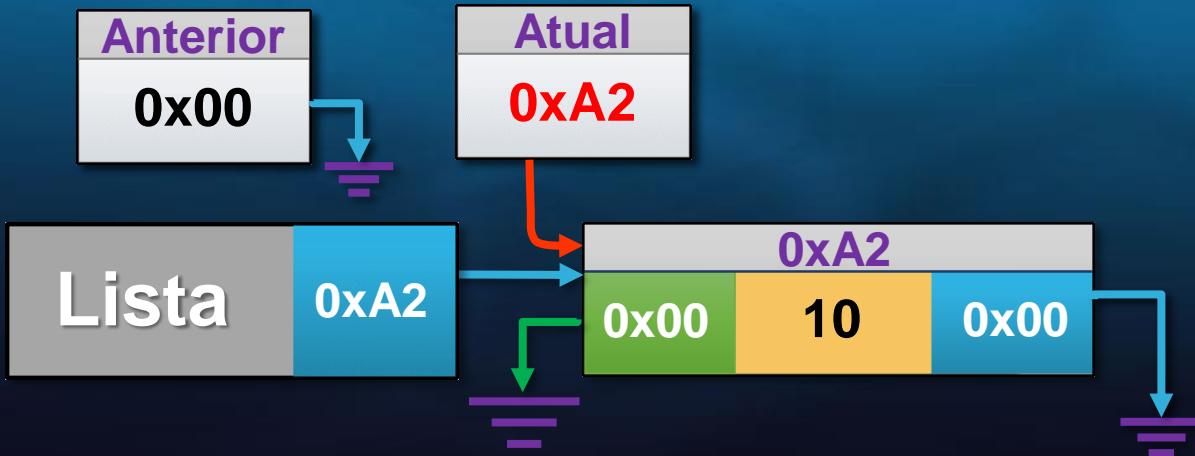
SIMULAÇÃO 1



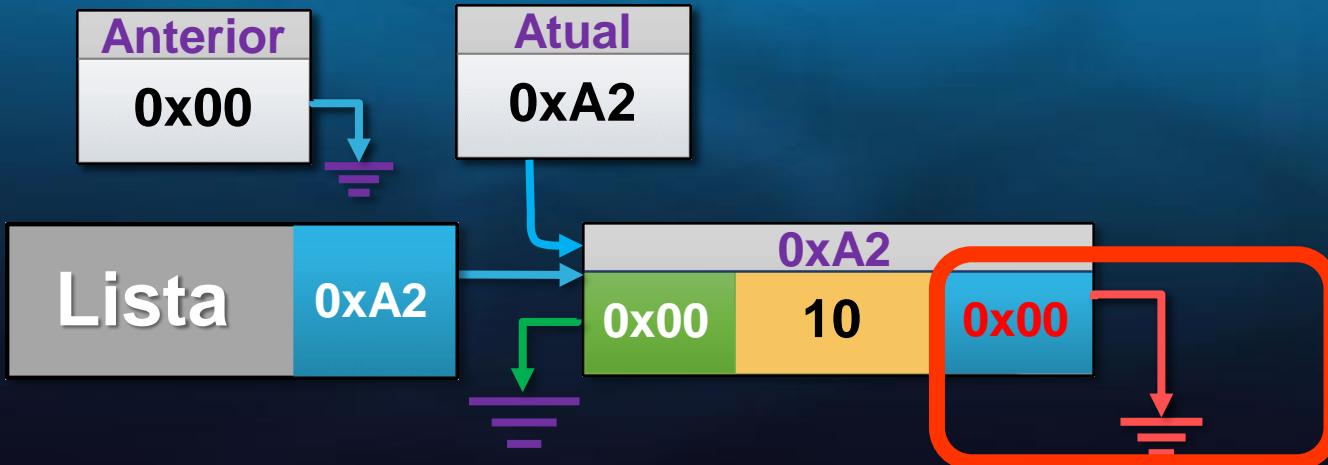
```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localiza o nó final da lista  
while (ptrNoAtual->proxNo != NULL) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```



```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localiza o nó final da lista  
while (ptrNoAtual->proxNo != NULL) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```

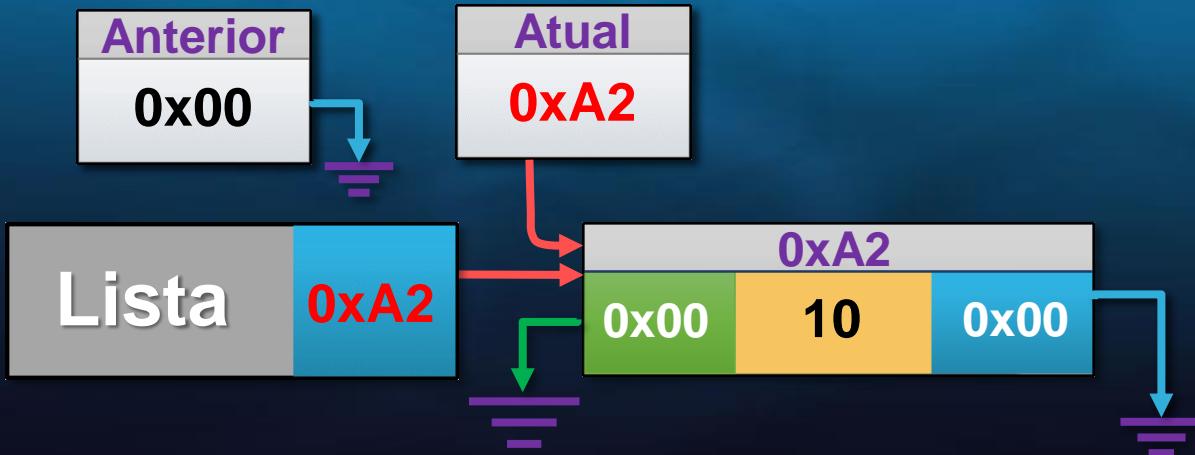


```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localiza o nó final da lista  
while (ptrNoAtual->proxNo != NULL) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```



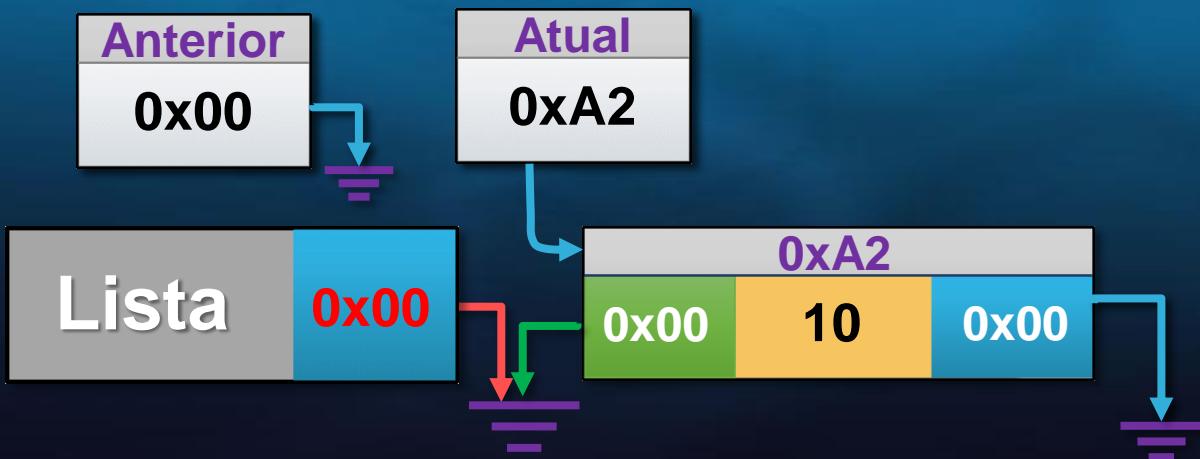
```
// Se for o primeiro nó da lista
if (ptrNoAtual == ptrLista->inicio) [
    ptrLista->inicio = NULL;
}
else {
    ptrNoAnterior->proxNo = NULL;
}

// Exclui o primeiro nó
delete ptrNoAtual;
```



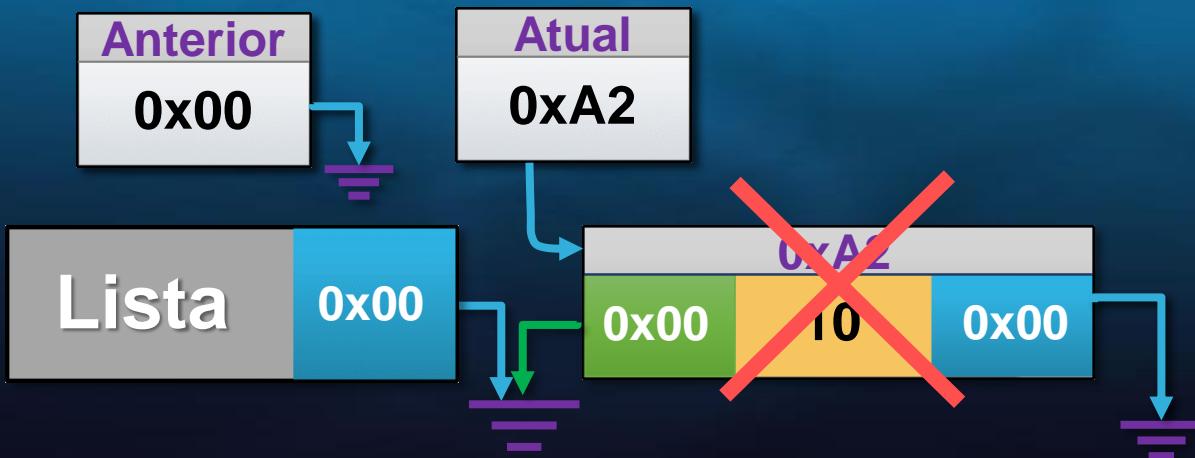
```
// Se for o primeiro nó da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrLista->inicio = NULL;
}
else {
    ptrNoAnterior->proxNo = NULL;
}

// Exclui o primeiro nó
delete ptrNoAtual;
```



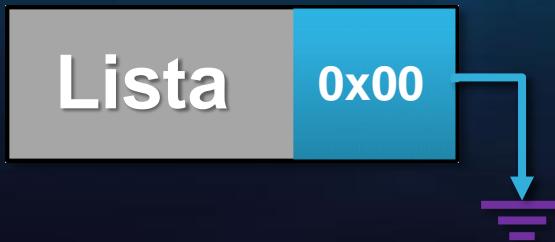
```
// Se for o primeiro nó da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrLista->inicio = NULL;
}
else {
    ptrNoAnterior->proxNo = NULL;
}

// Exclui o primeiro nó
delete ptrNoAtual;
```



```
// Se for o primeiro nó da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrLista->inicio = NULL;
}
else {
    ptrNoAnterior->proxNo = NULL;
}

// Exclui o primeiro nó
delete ptrNoAtual;
```

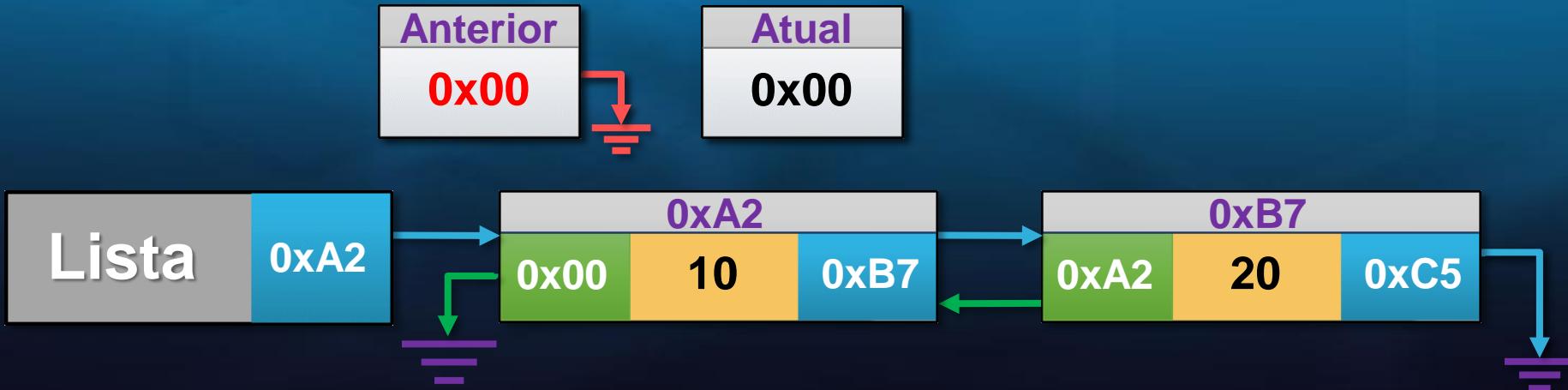


EXCLUIR NO FINAL DA LISTA

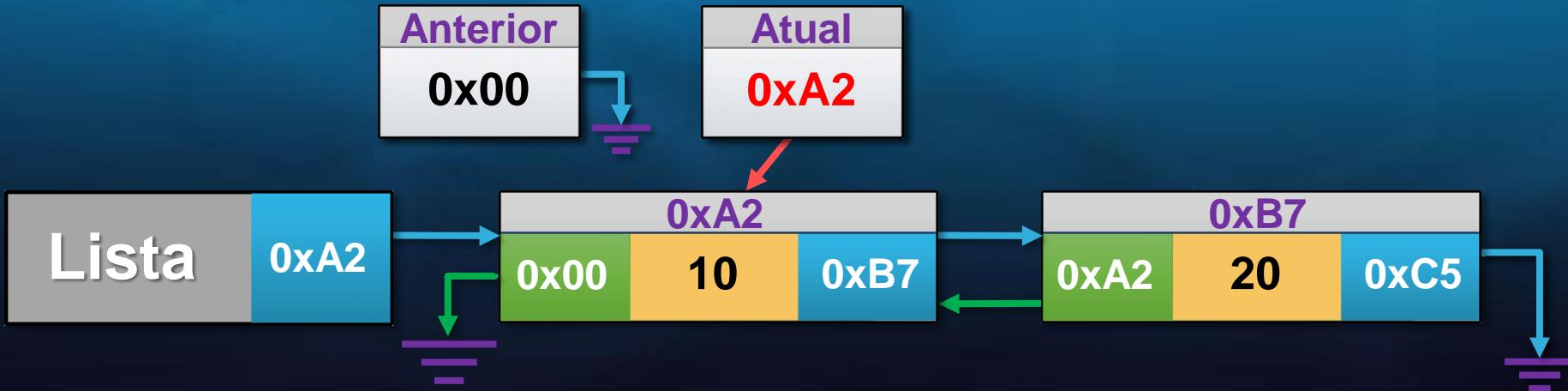
SIMULAÇÃO 2



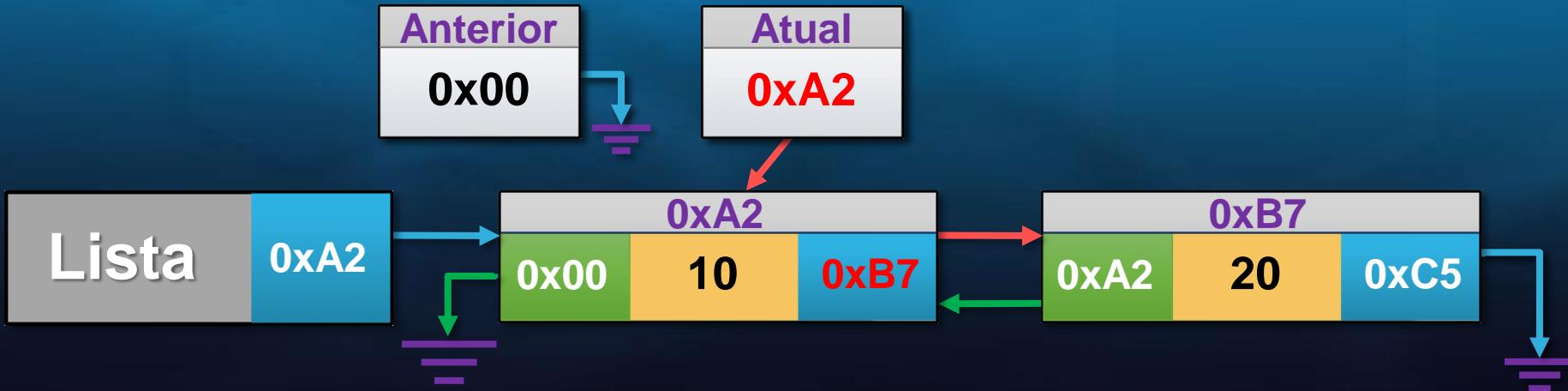
```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localiza o nó final da lista  
while (ptrNoAtual->proxNo != NULL) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```



```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localiza o nó final da lista  
while (ptrNoAtual->proxNo != NULL) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```



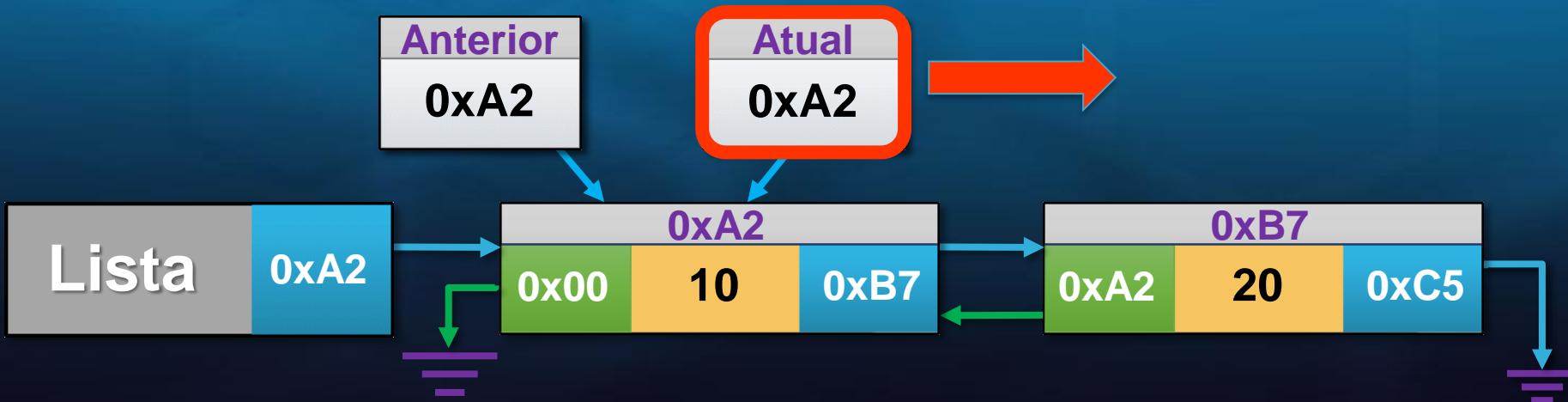
```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localiza o nó final da lista  
while (ptrNoAtual->proxNo != NULL) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```



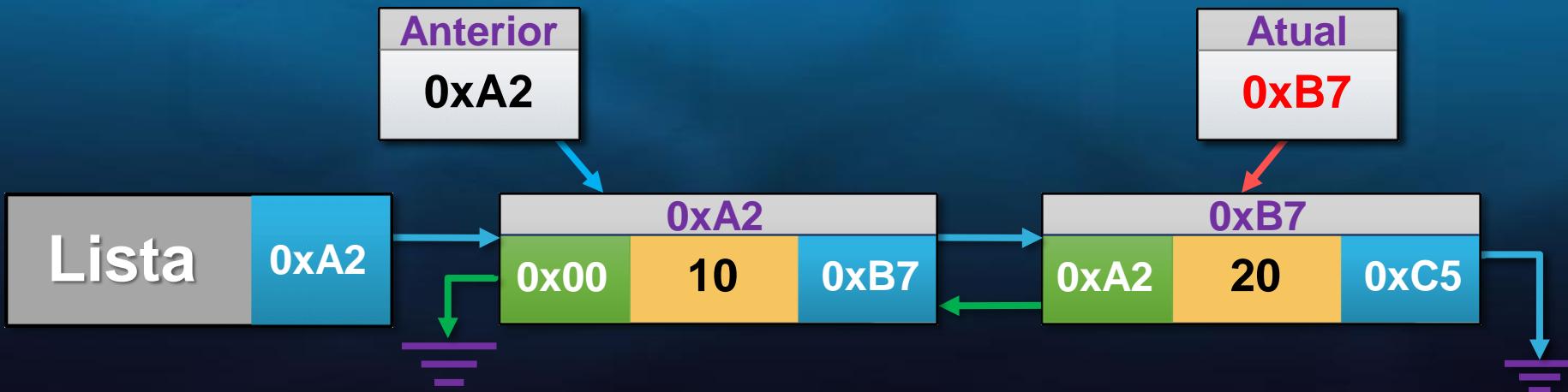
```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localiza o nó final da lista  
while (ptrNoAtual->proxNo != NULL) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```



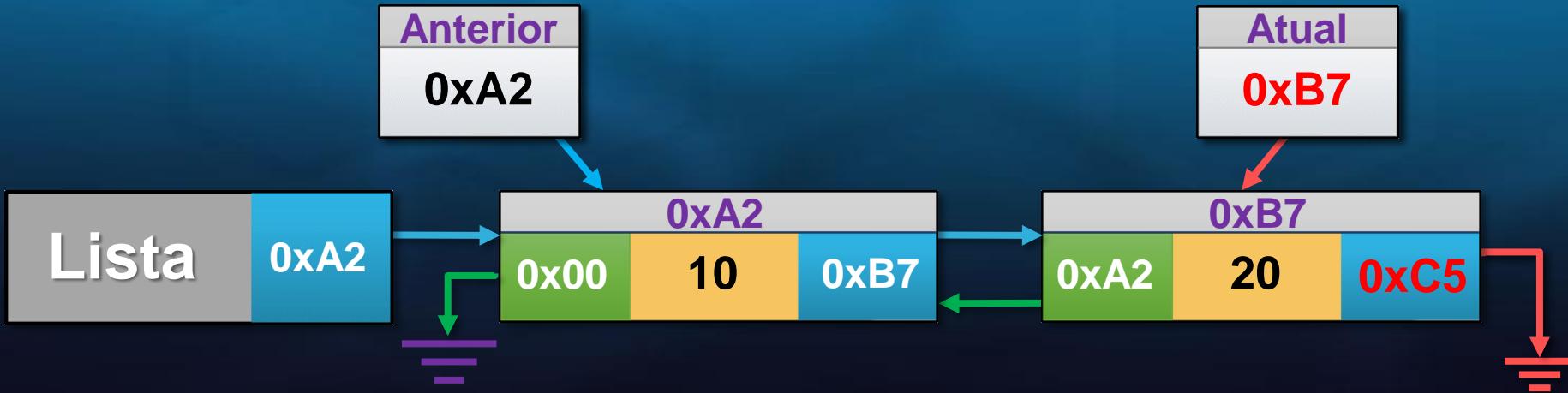
```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localiza o nó final da lista  
while (ptrNoAtual->proxNo != NULL) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```



```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localiza o nó final da lista  
while (ptrNoAtual->proxNo != NULL) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```

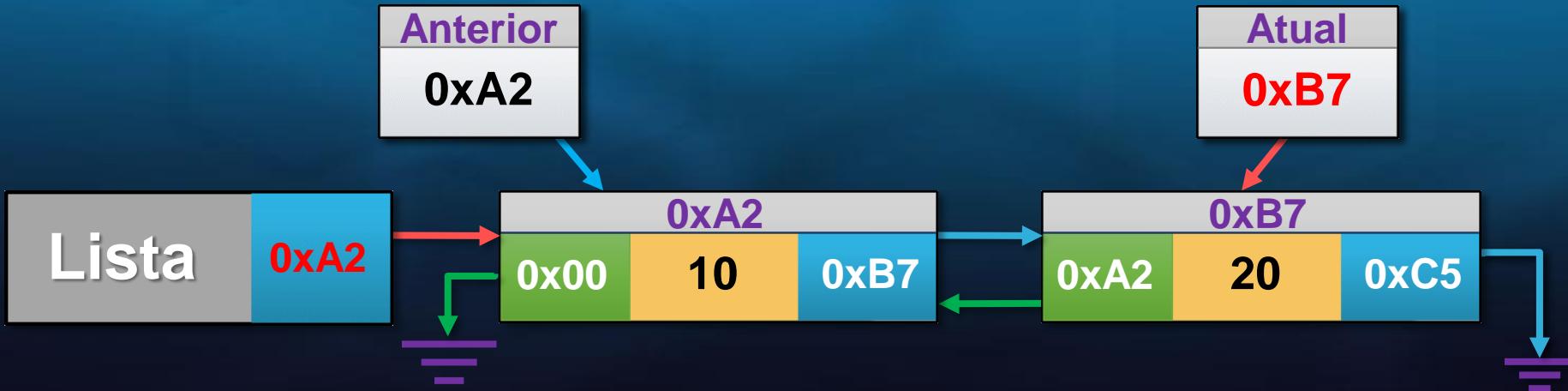


```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localiza o nó final da lista  
while (ptrNoAtual->proxNo != NULL) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```



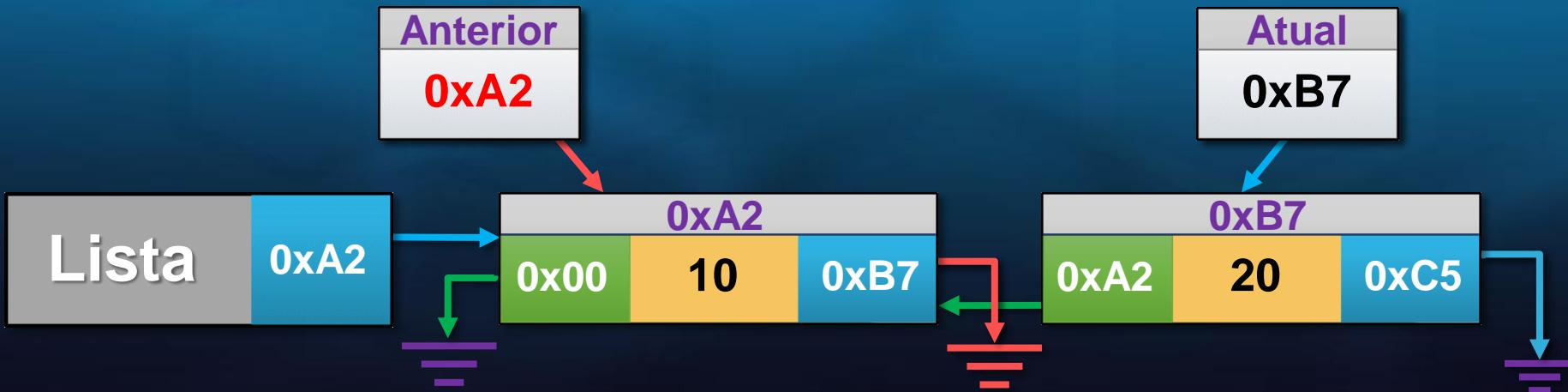
```
// Se for o primeiro nó da lista
if (ptrNoAtual == ptrLista->inicio) [
    ptrLista->inicio = NULL;
}
else {
    ptrNoAnterior->proxNo = NULL;
}

// Exclui o primeiro nó
delete ptrNoAtual;
```



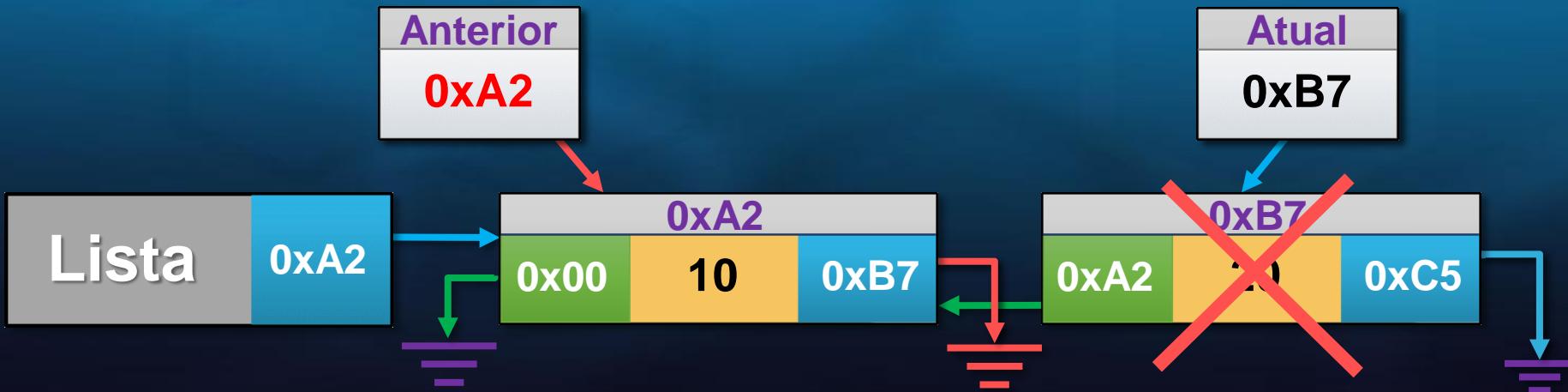
```
// Se for o primeiro nó da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrLista->inicio = NULL;
}
else {
    ptrNoAnterior->proxNo = NULL;
}

// Exclui o primeiro nó
delete ptrNoAtual;
```



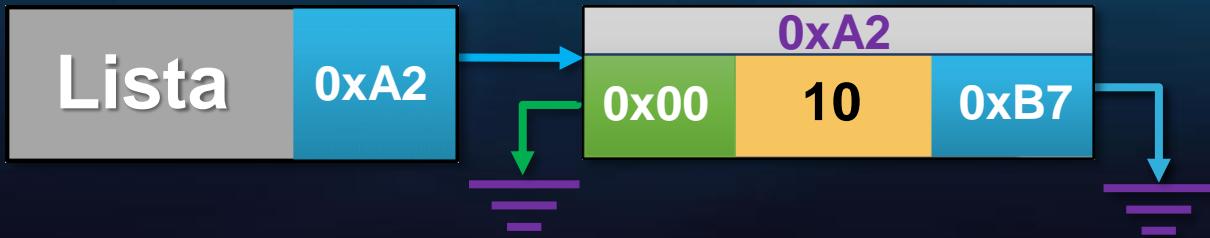
```
// Se for o primeiro nó da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrLista->inicio = NULL;
}
else {
    ptrNoAnterior->proxNo = NULL;
}

// Exclui o primeiro nó
delete ptrNoAtual;
```



```
// Se for o primeiro nó da lista
if (ptrNoAtual == ptrLista->inicio) {
    ptrLista->inicio = NULL;
}
else {
    ptrNoAnterior->proxNo = NULL;
}

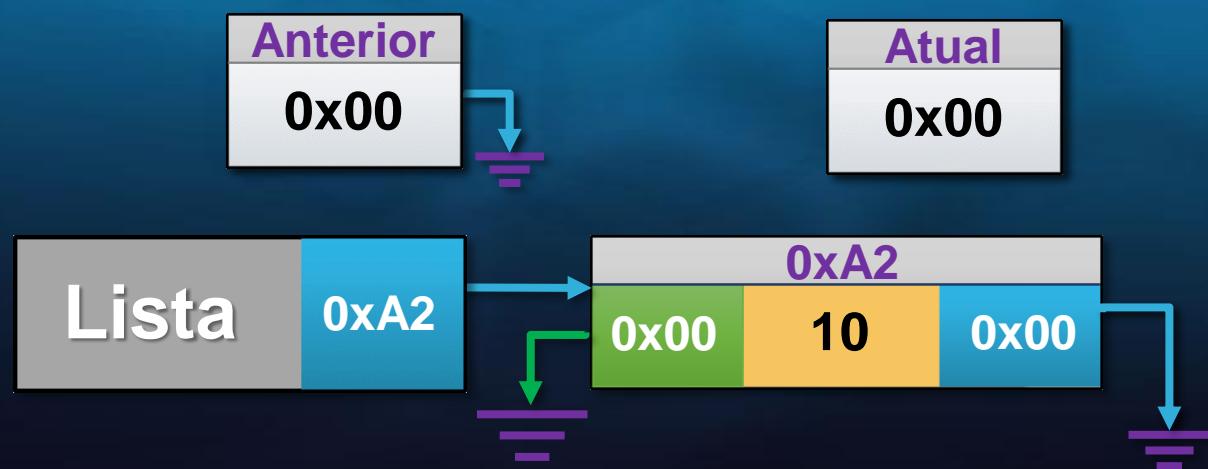
// Exclui o primeiro nó
delete ptrNoAtual;
```



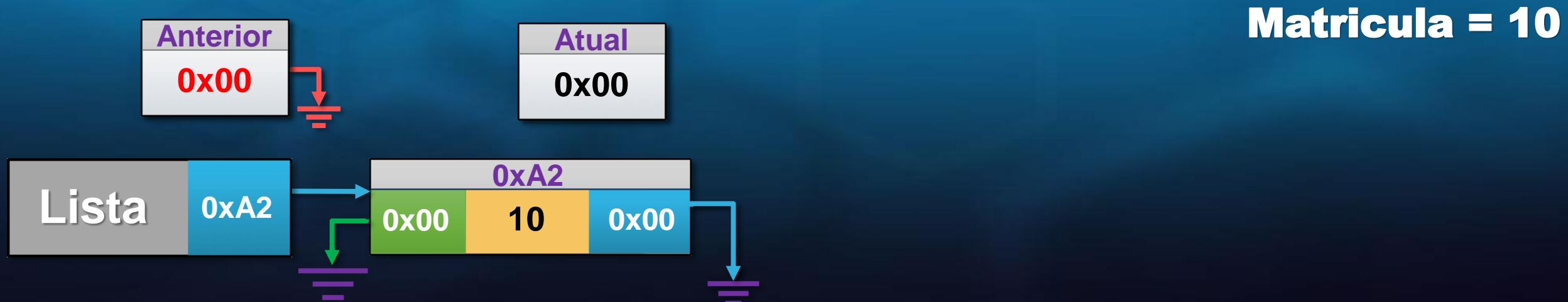
EXCLUIR ORDENADO NA LISTA

SIMULAÇÃO 1

Matricula = 10



```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;  
  
// Localizao nó que será excluído  
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}  
  
if (ptrNoAtual == NULL) {  
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;  
    return false;  
}
```



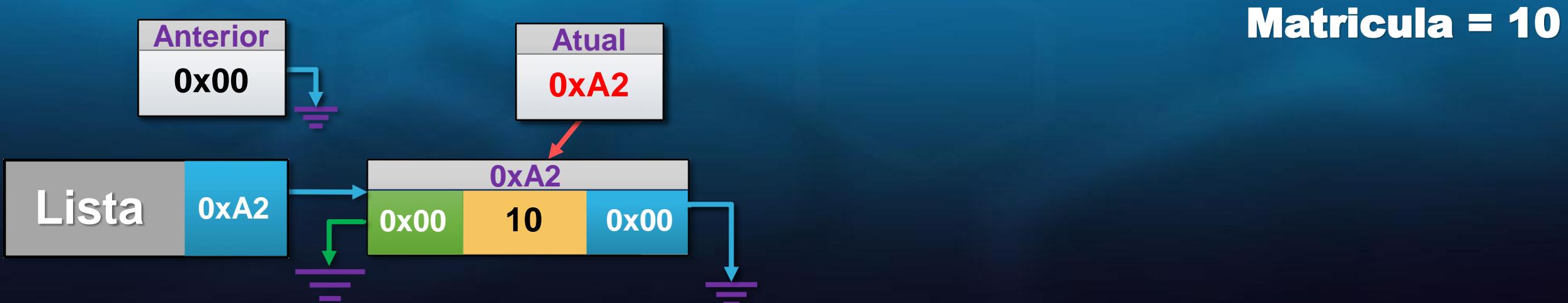
```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```



```

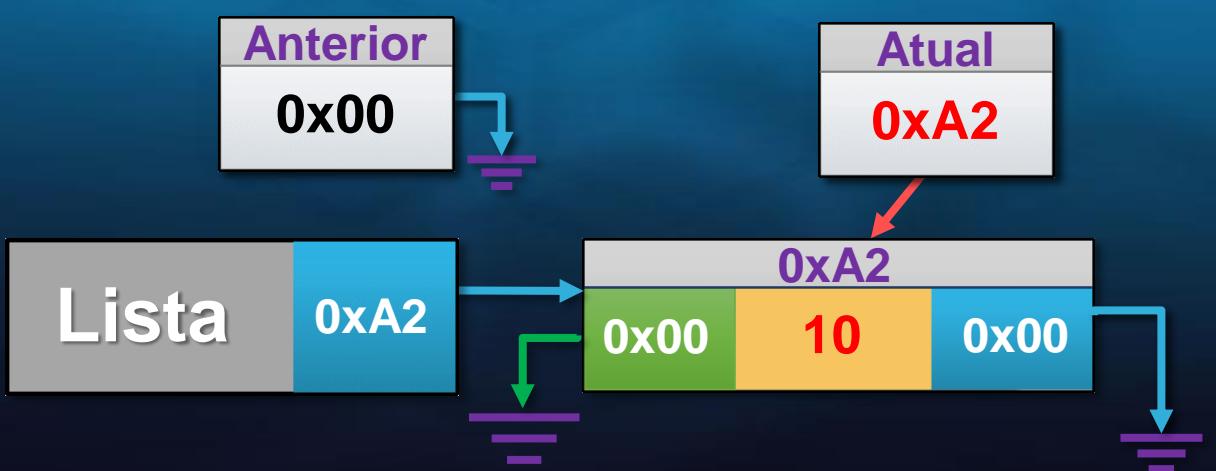
ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localiza o nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```

$$10 \neq 10$$



Matricula = 10

```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```



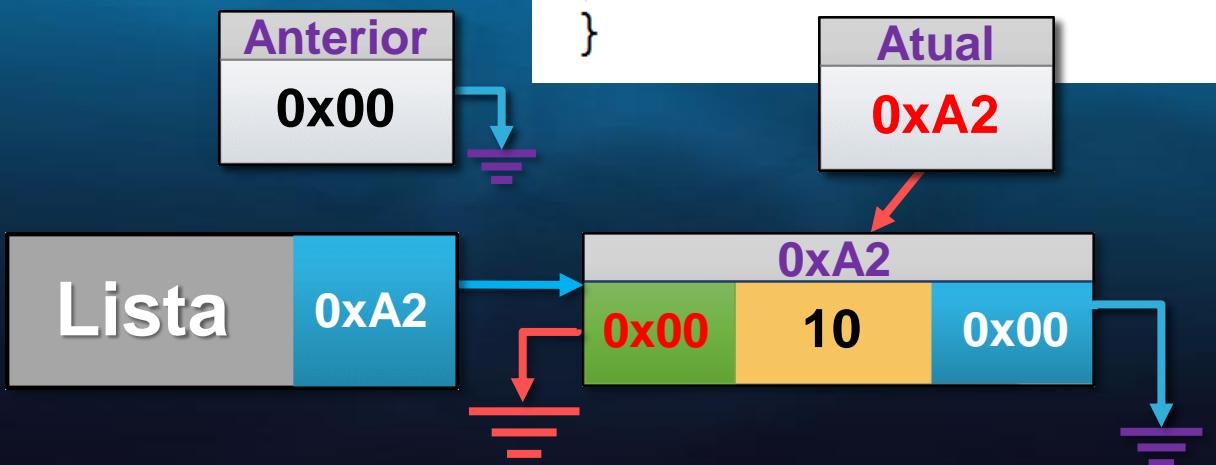
```

// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = NULL;
    }
}
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

```



Matricula = 10

```

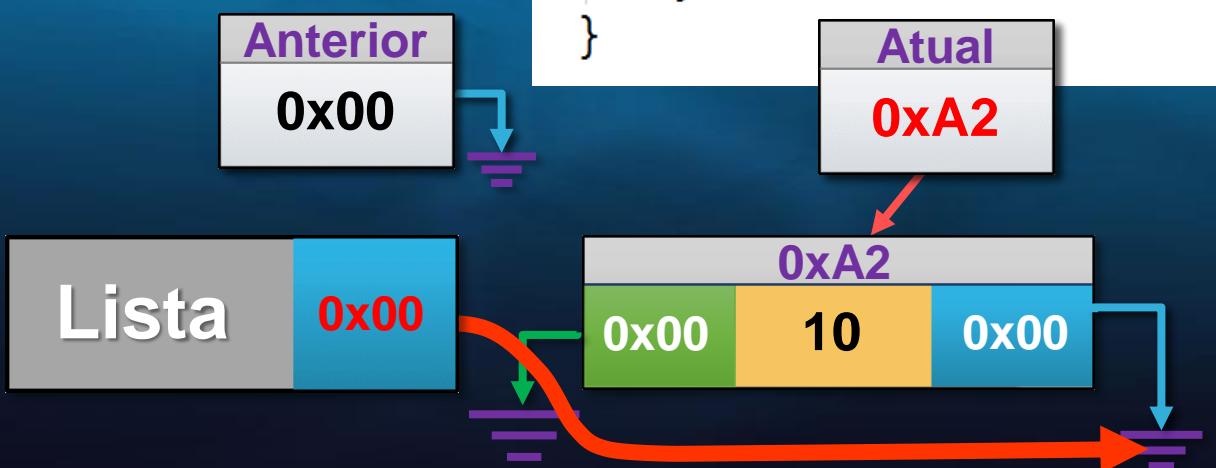
// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;
}

// Se houver mais que um nó na lista
if (ptrNoAtual->proxNo != NULL) {
    ptrNoAtual->proxNo->antNo = NULL;
}
}

else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}
}

```

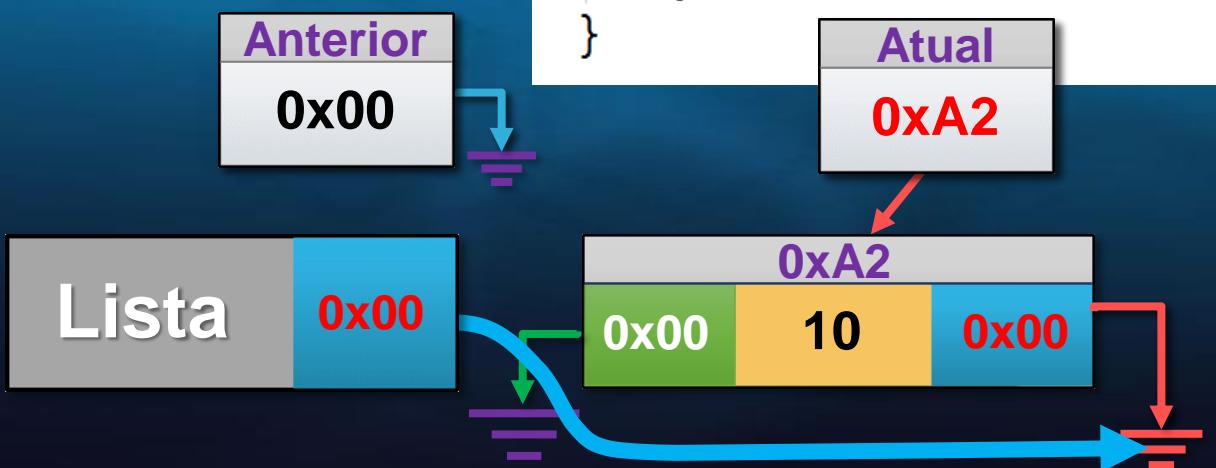


Matricula = 10

```
// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = NULL;
    }
}
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}
```

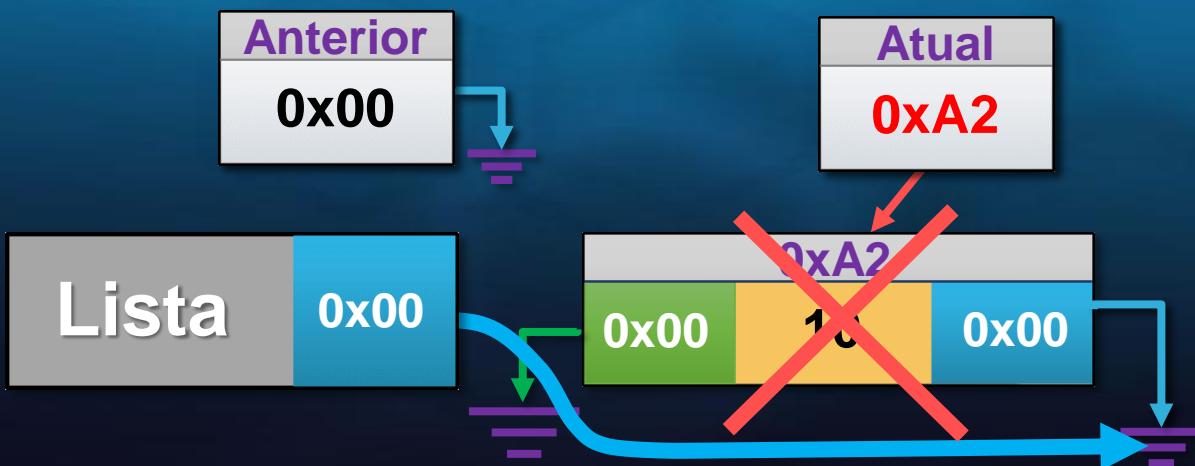


Matricula = 10

```
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

// Exclui o nó atual
delete ptrNoAtual;
```



Matricula = 10

```
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

// Exclui o nó atual
delete ptrNoAtual;
```

Lista

0x00



EXCLUIR ORDENADO NA LISTA

SIMULAÇÃO 2

Matricula = 10



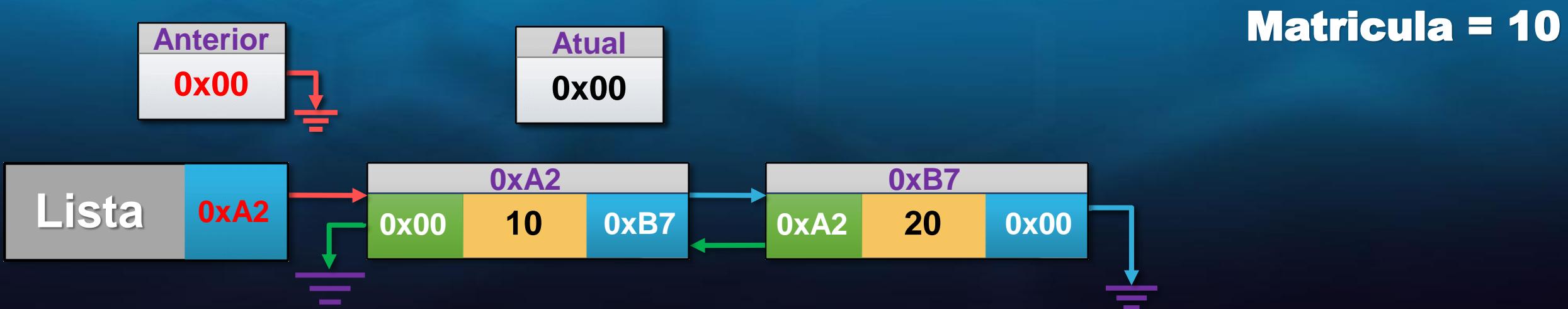
```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```



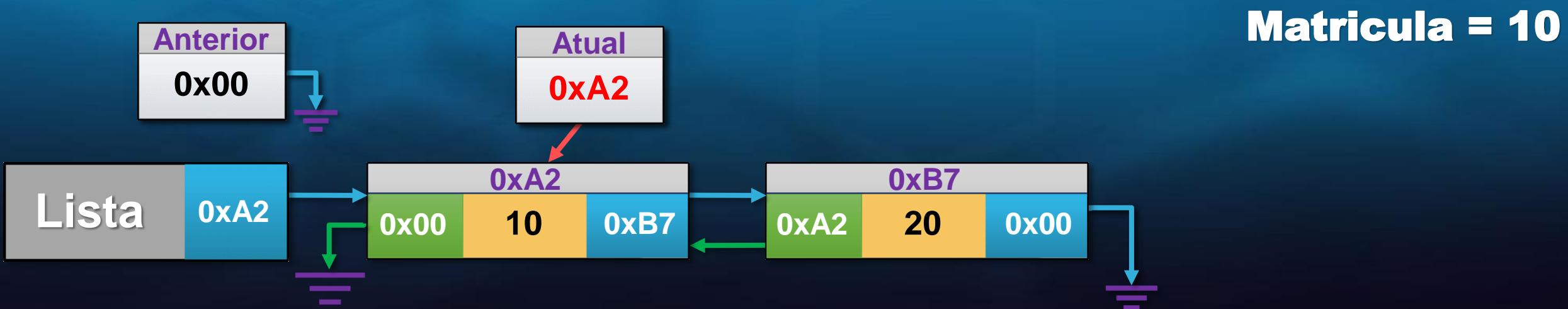
```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```

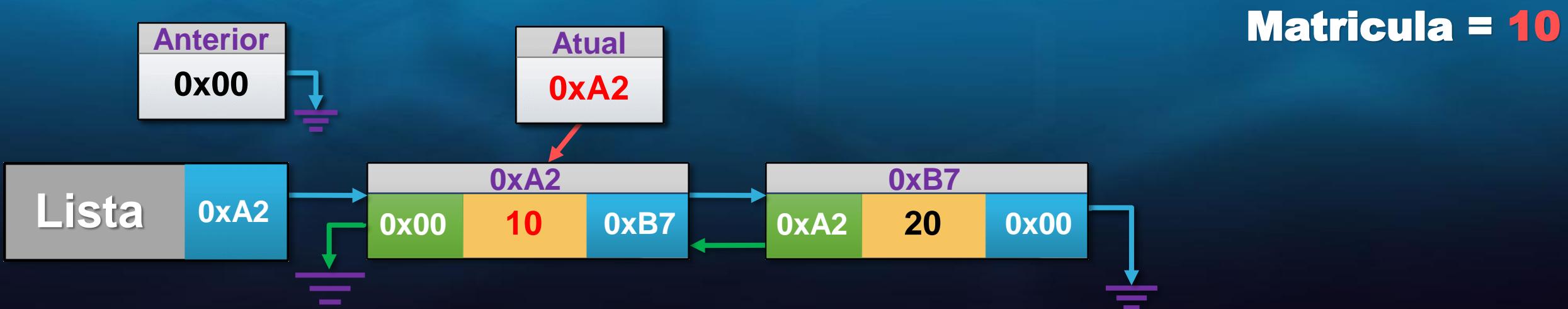


```
ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não existe."
    return false;
}
```

$$10 \neq 10$$



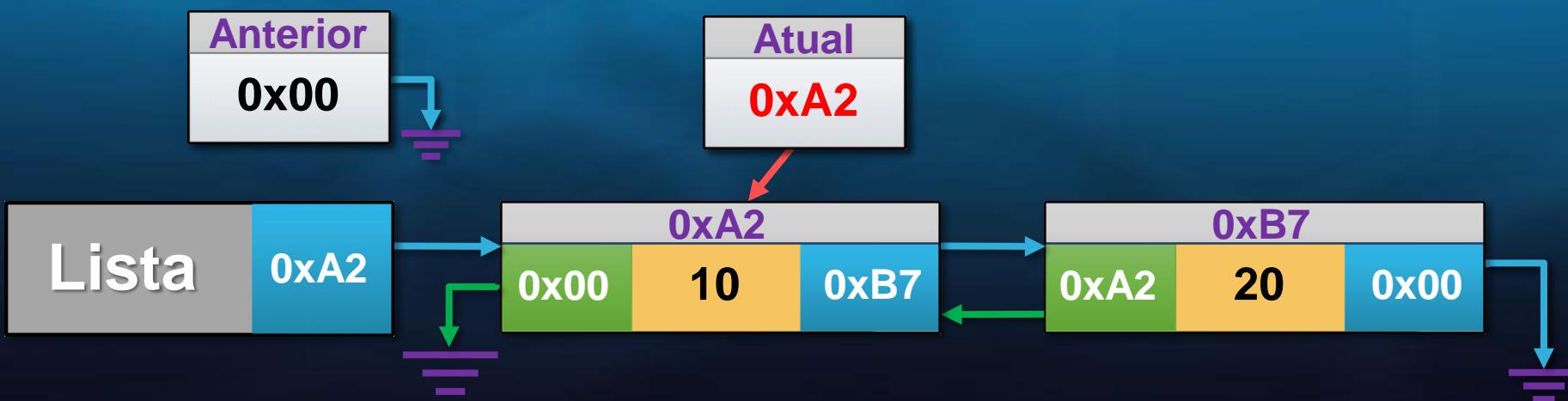
```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

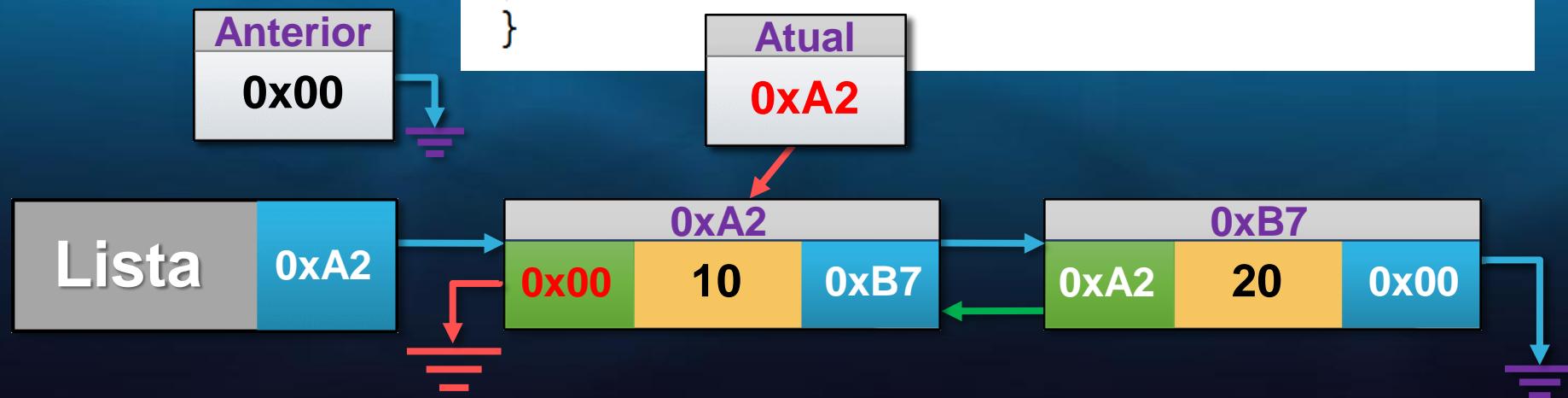
// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```



```
// Se for o primeiro nó da lista  
if (ptrNoAtual->antNo == NULL) {  
    ptrLista->inicio = ptrNoAtual->proxNo;  
  
    // Se houver mais que um nó na lista  
    if (ptrNoAtual->proxNo != NULL) {  
        ptrNoAtual->proxNo->antNo = NULL;  
    }  
}  
else {  
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;  
  
    // Se houver mais que um nó na lista  
    if (ptrNoAtual->proxNo != NULL) {  
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;  
    }  
}
```



```

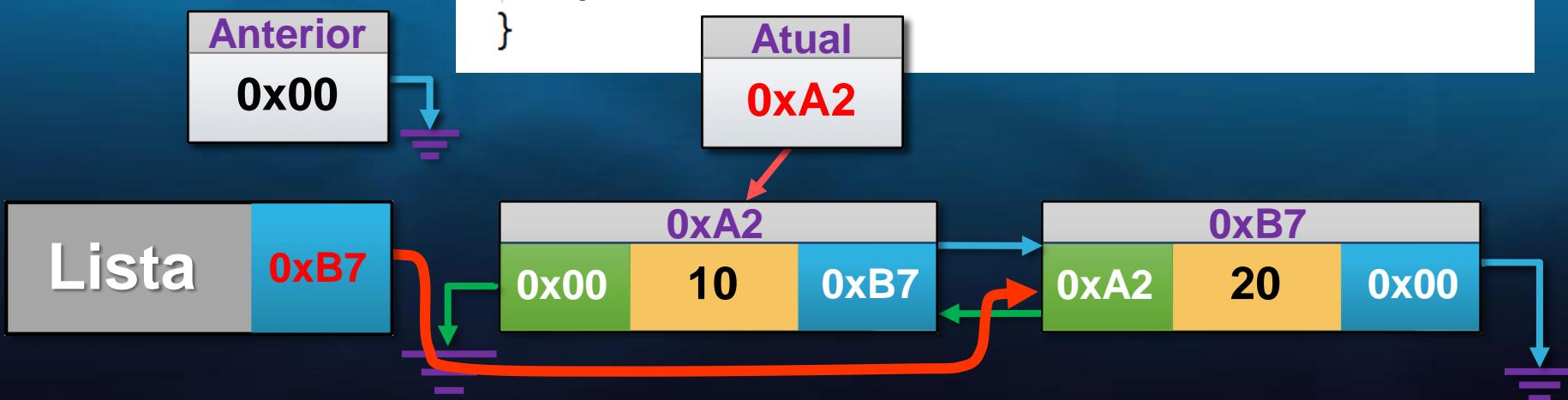
// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;
}

// Se houver mais que um nó na lista
if (ptrNoAtual->proxNo != NULL) {
    ptrNoAtual->proxNo->antNo = NULL;
}
}

else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}
}

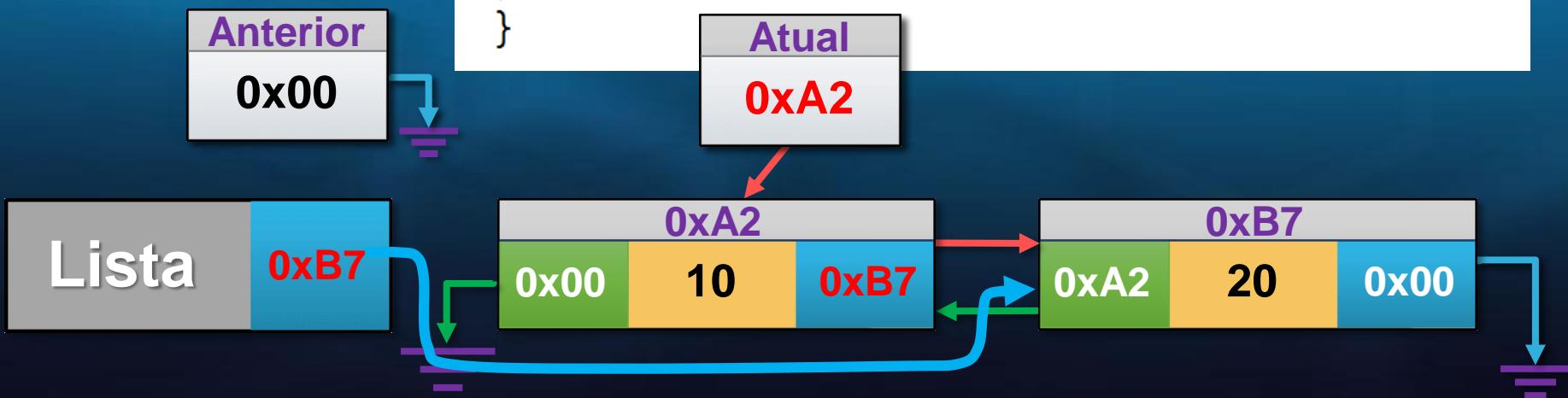
```



```
// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = NULL;
    }
}
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

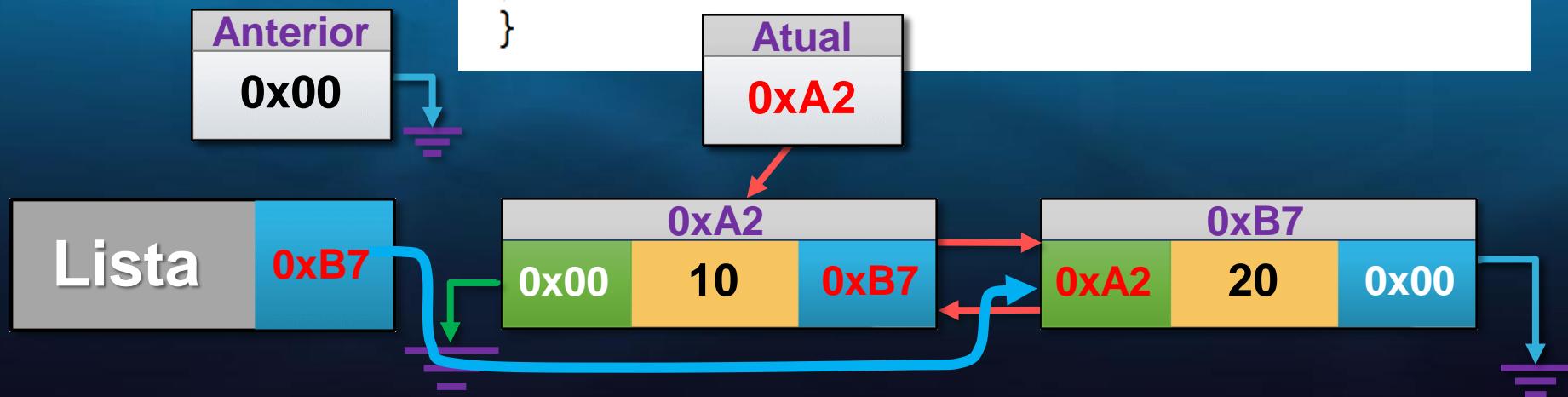
    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}
```



```
// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = NULL;
    }
}
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}
```



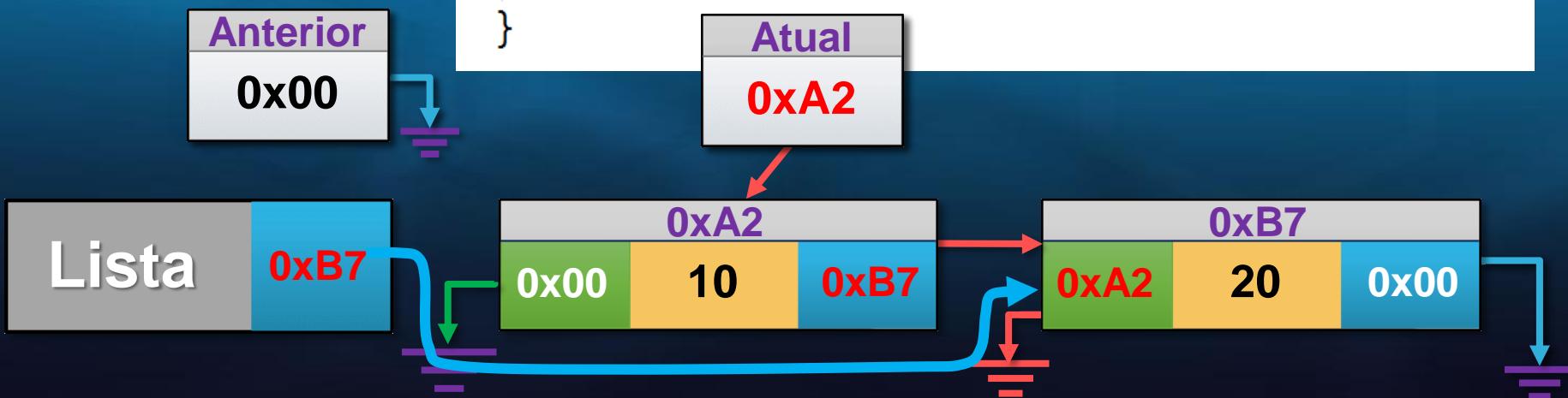
```

// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = NULL;
    }
}
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

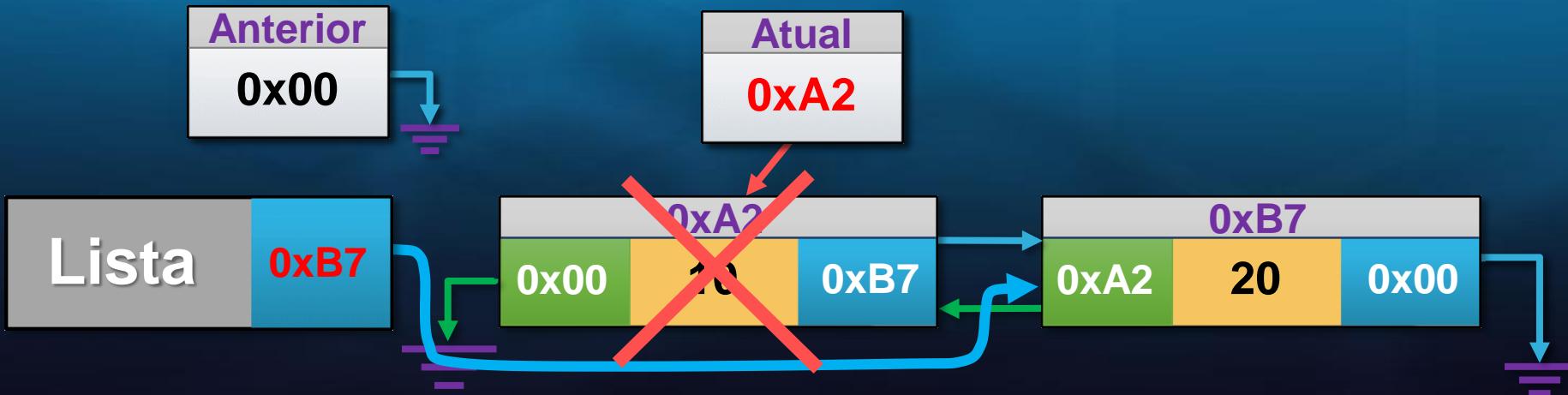
```



```
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

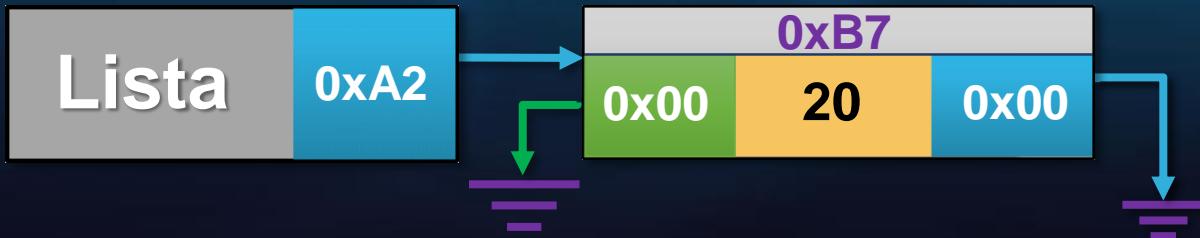
// Exclui o nó atual
delete ptrNoAtual;
```



```
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

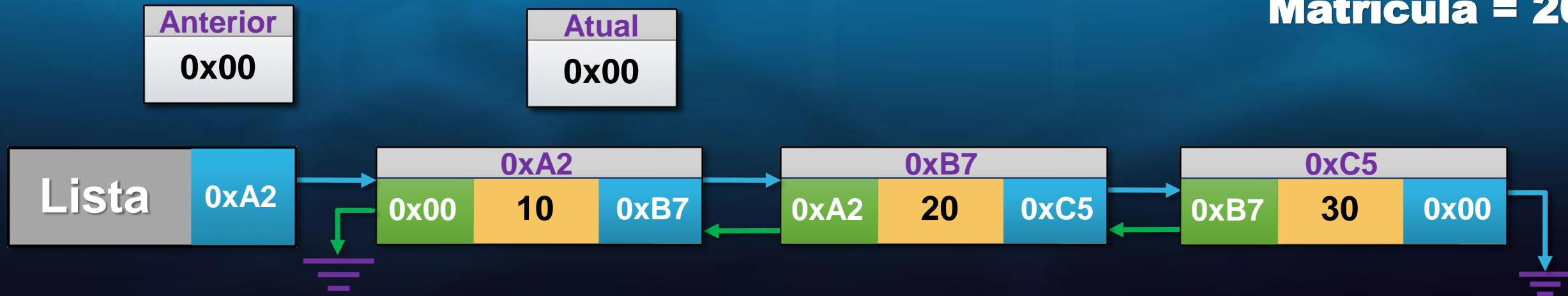
// Exclui o nó atual
delete ptrNoAtual;
```



EXCLUIR ORDENADO NA LISTA

SIMULAÇÃO 3

Matricula = 20



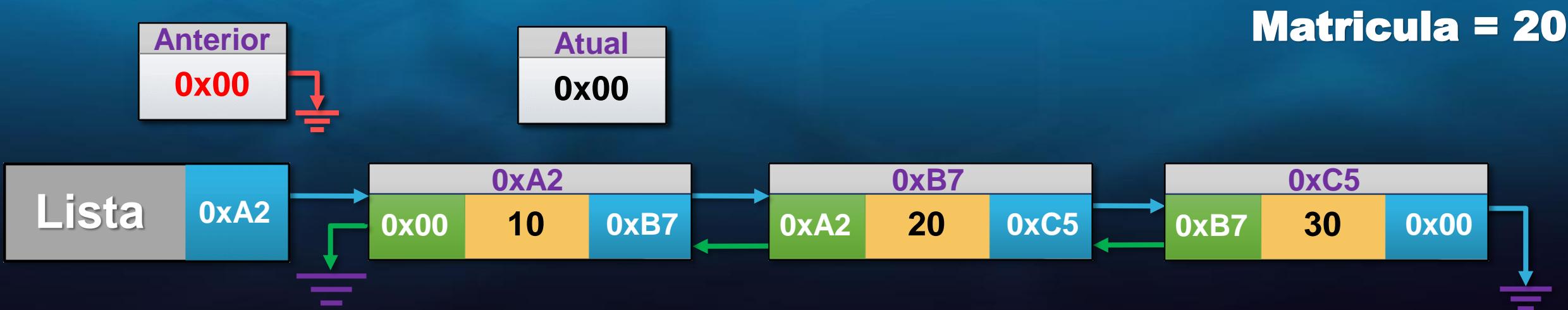
```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```



```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```



```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;
```

// Localiza o nó que será excluído

```
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```

```
if (ptrNoAtual == NULL) {  
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;  
    return false;  
}
```

10 != 20



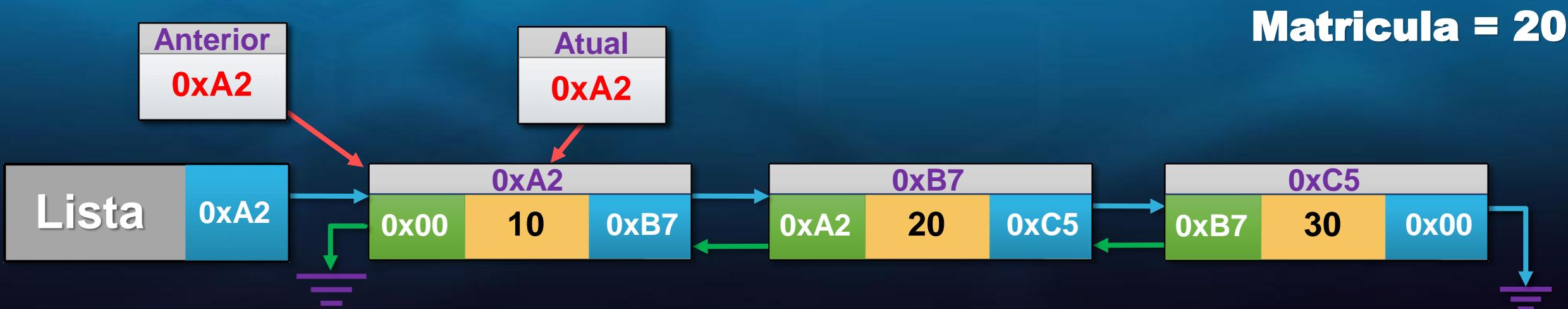
```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```



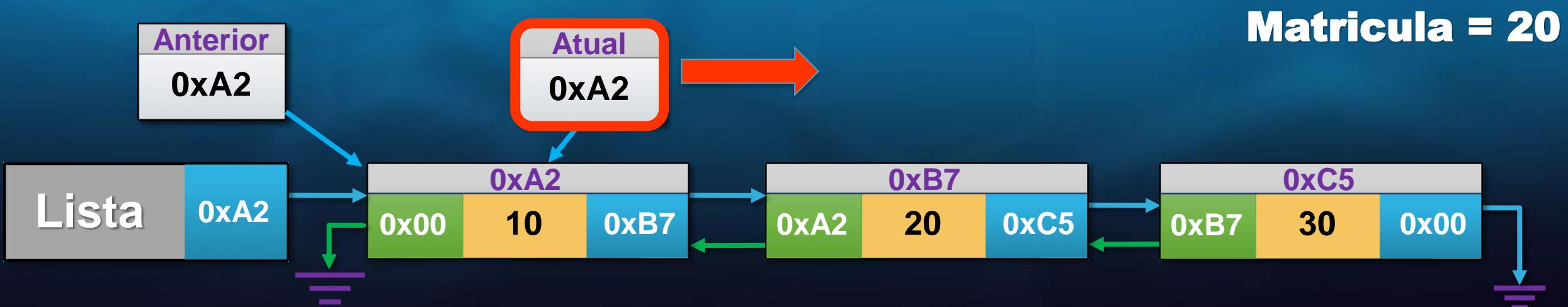
```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```



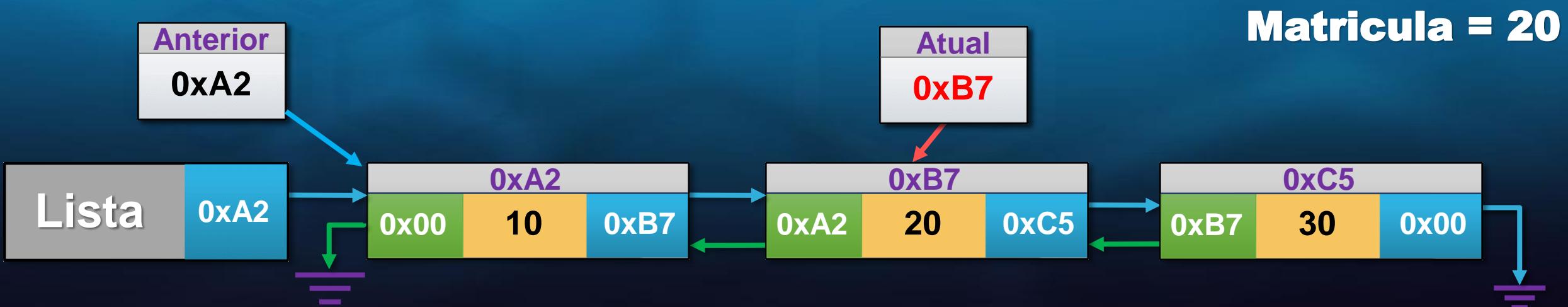
```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```



```
ptrNoAnterior = NULL;  
ptrNoAtual = ptrLista->inicio;
```

```
// Localizao nó que será excluído
```

```
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {  
    ptrNoAnterior = ptrNoAtual;  
    ptrNoAtual = ptrNoAtual->proxNo;  
}
```

```
if (ptrNoAtual == NULL) {  
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;  
    return false;  
}
```

20 != 20



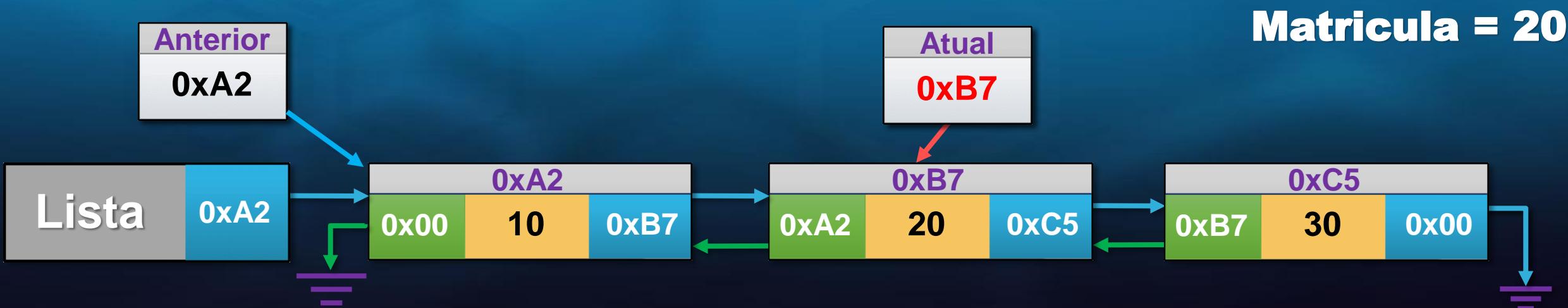
```

ptrNoAnterior = NULL;
ptrNoAtual = ptrLista->inicio;

// Localizao nó que será excluído
while (ptrNoAtual != NULL && ptrNoAtual->dadosAluno.matricula != matricula) {
    ptrNoAnterior = ptrNoAtual;
    ptrNoAtual = ptrNoAtual->proxNo;
}

if (ptrNoAtual == NULL) {
    cout << "A matrícula " << matricula << " não foi encontrada!" << endl;
    return false;
}

```



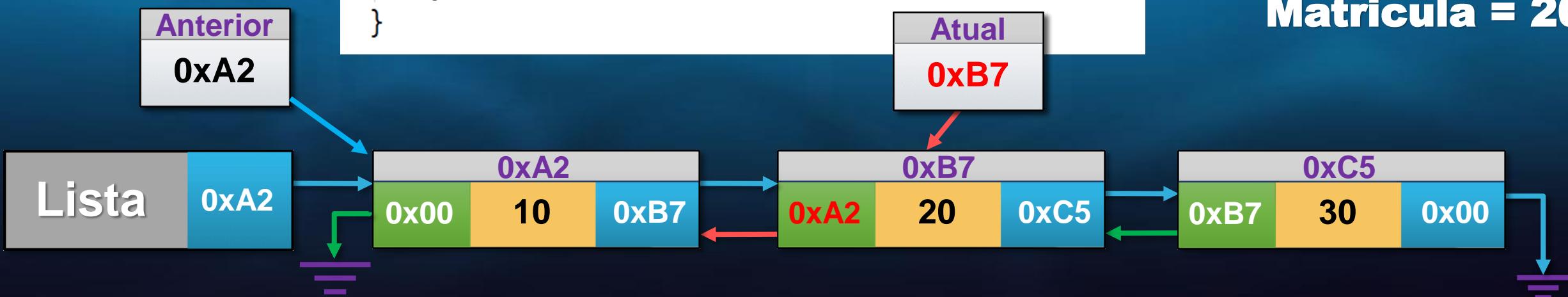
```

// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = NULL;
    }
}
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

```



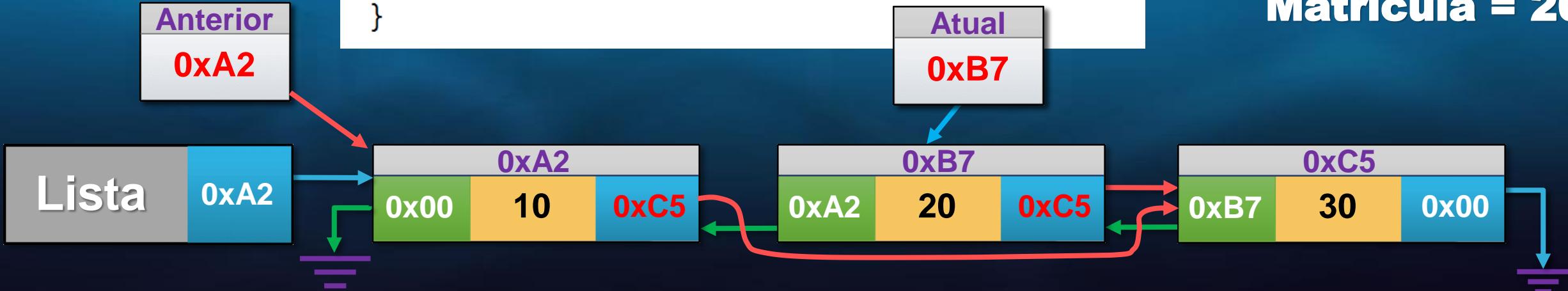
```

// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = NULL;
    }
}
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;
}

// Se houver mais que um nó na lista
if (ptrNoAtual->proxNo != NULL) {
    ptrNoAtual->proxNo->antNo = ptrNoAnterior;
}
}

```



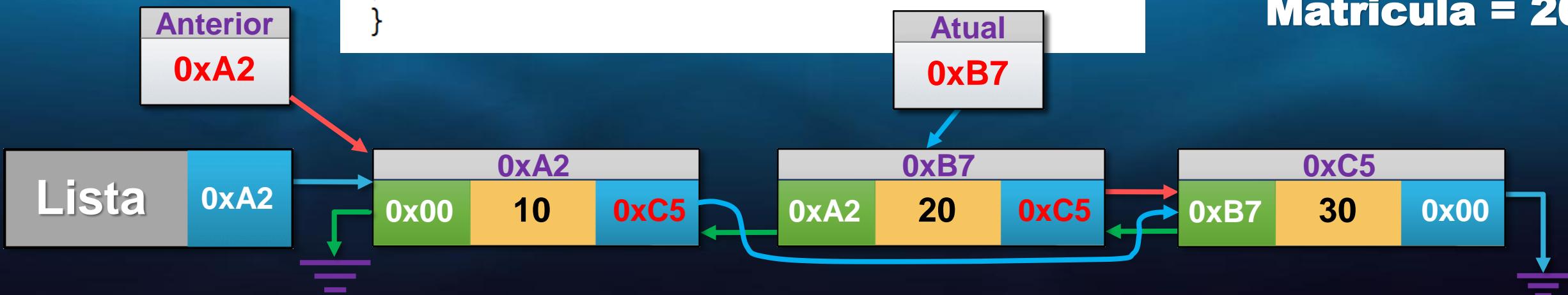
```

// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = NULL;
    }
}
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

```



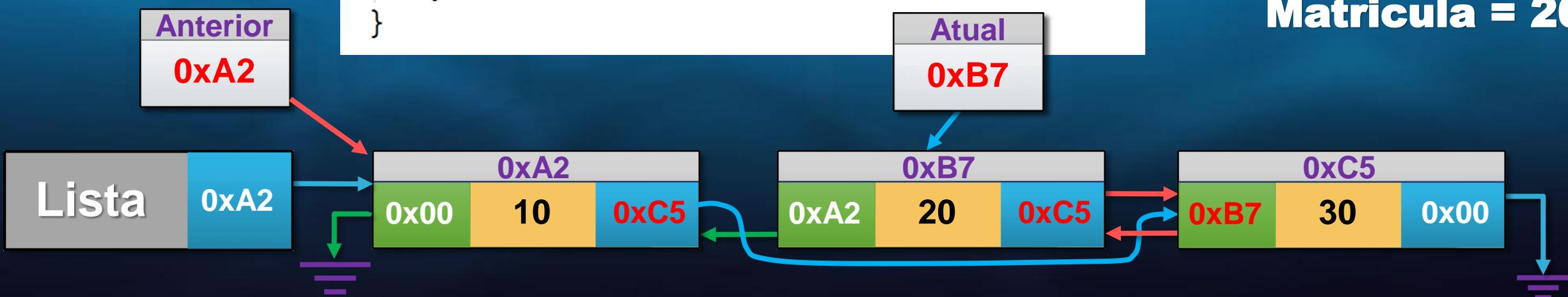
```

// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = NULL;
    }
}
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

```



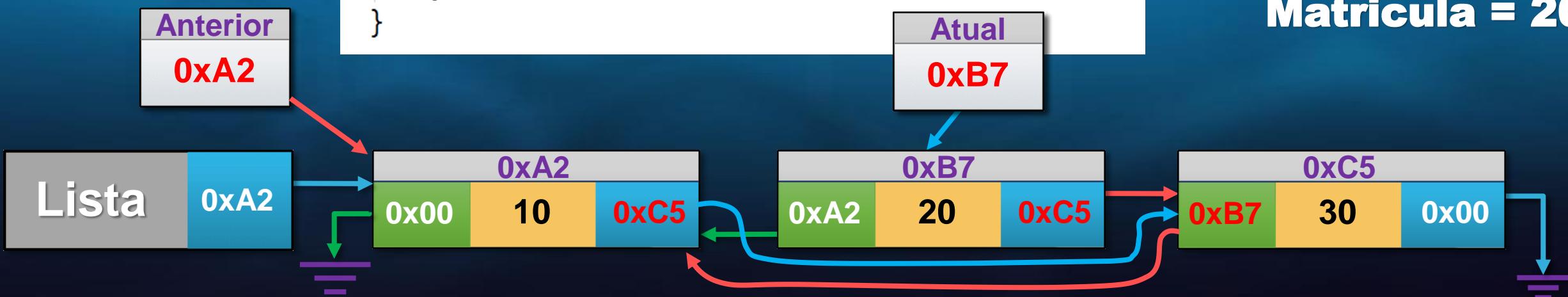
```

// Se for o primeiro nó da lista
if (ptrNoAtual->antNo == NULL) {
    ptrLista->inicio = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = NULL;
    }
}
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

```



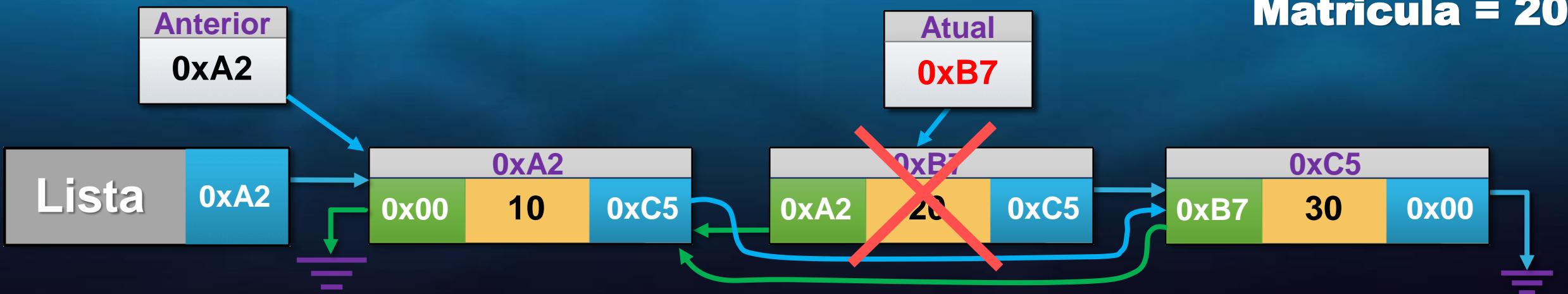
```

else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

// Exclui o nó atual
delete ptrNoAtual;

```



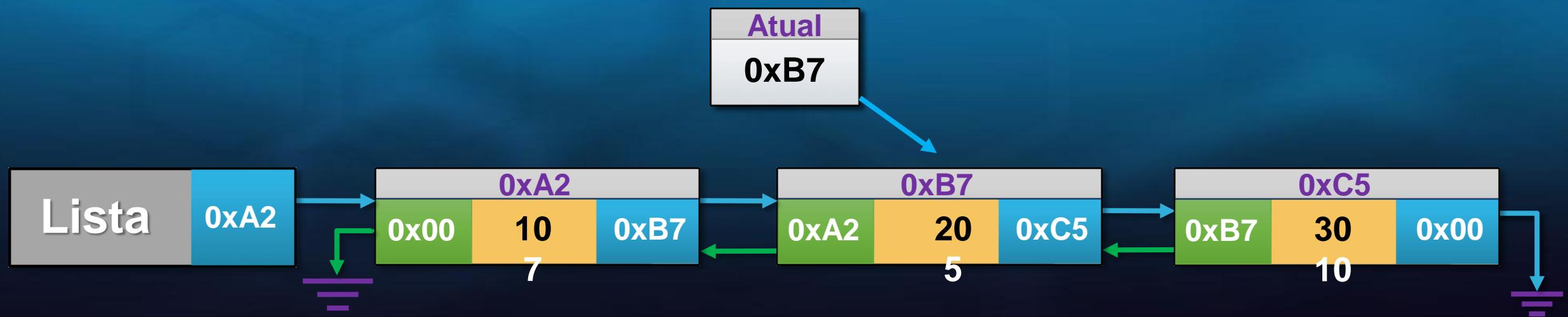
```
else {
    ptrNoAnterior->proxNo = ptrNoAtual->proxNo;

    // Se houver mais que um nó na lista
    if (ptrNoAtual->proxNo != NULL) {
        ptrNoAtual->proxNo->antNo = ptrNoAnterior;
    }
}

// Exclui o nó atual
delete ptrNoAtual;
```

Matricula = 20





Exercício – A01

- Insira, via programação, 5 alunos a uma lista.
- Em seguida, crie uma função que imprima os dados em **ordem reversa**, ou seja, do último para o primeiro.

Exercício – A02

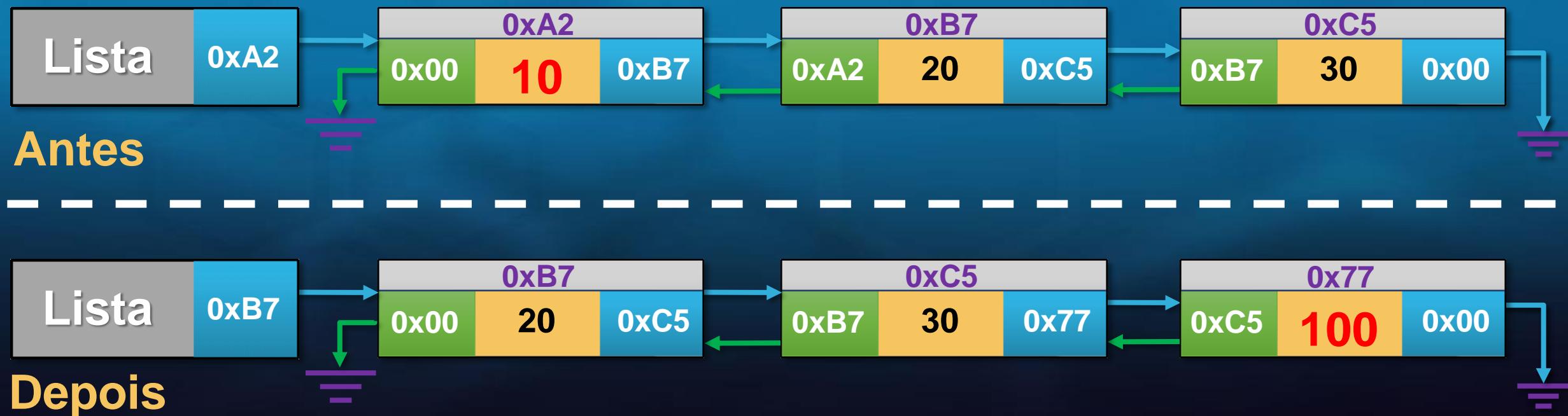
- Crie 3 lista (**ListaA**, **ListaB** e **ListaC**).
- Informe 3 alunos diferentes, tanto para **ListaA** quanto para **ListaB**.
- Concatene a **ListaA** e **ListaB** na **ListaC**.
- Após a função, imprima a lista **ListaC**.

Exercício – A03

- Adicione o campo **sexo** a estrutura de Dados que deverá receber **F** ou **M** de acordo com o aluno.
- Crie **duas lista**, uma para guardar dados dos alunos **masculino** e outra para o **feminino**.
- Peça na tela que o **usuário** informe dados **de 6 alunos**.
 - **Obs: usar cin.**
- A cada novo aluno informado, insira os dados na respectiva lista de acordo com o sexo.
- Por fim, **exiba as duas listas na tela**.

Exercício – A04

- Crie uma função que altere o número da matrícula.
- A função deverá receber o **número da matrícula** a ser alterado e o novo valor. Além da alteração, a função deverá garantir a ordenação da lista conforme o exemplo abaixo.
- **Exemplo: Alterar a matrícula de 10 para 100.**

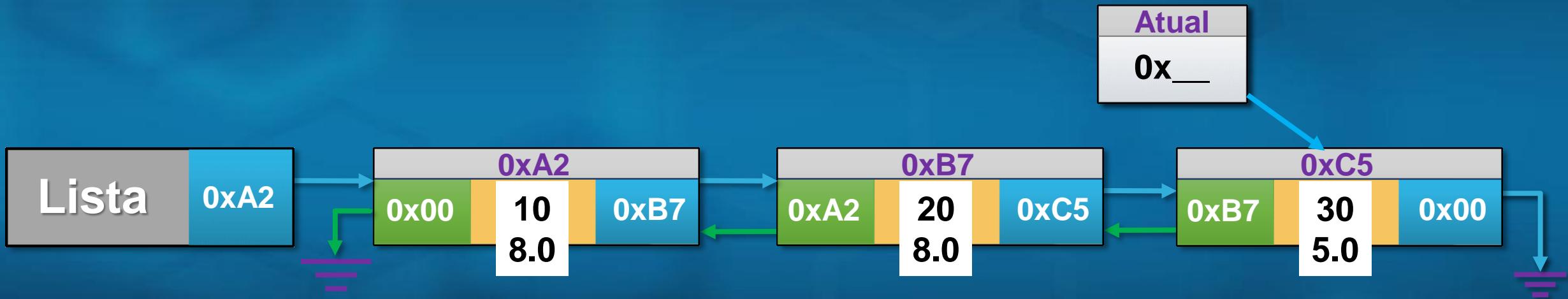


Exercício – A05

- Insira 5 alunos em uma lista.
- Faça uma função que percorra a lista, do início até o final procurando a **menor média**.
- Chegando ao final, volte até o nó com a menor média, **imprima seus dados, os do nó anterior e do próximo**.
- **OBS 1:** para simplificar o programa, garanta que exista apenas uma menor média.
- **OBS 2:** dar uma especial atenção quando a menor média for no primeiro ou no último nó da lista.

Exercício – A05

- Exemplo



Pensamento

"O pensamento lógico pode levar você de A a B, mas a imaginação te leva a qualquer parte do Universo."

(Albert Einstein)

FIM

Prof. Dr Ricardo Luis Balieiro