Lista Duplamente Encadeada

Henrique C. Oliveira[1], Fernando P. Goularti[2], João Vitor S. Pivato[3], Pablo Benteu[4]

[1]Instituto de Matemática e Computacção – Universidade Federal de Itajubá (UNIFEI)
Caixa Postal 37500-903 – Minas Gerais – MG – Brazil

henriquecastro@unifei.edu.br, d2019017937@unifei.edu.br

d2019017900@unifei.edu.br, d2016012928@unifei.edu.br

Abstract. *This report presents a doubly linked list written in the "c" programming language, as its implementation. The program was implemented using an abstract data type (ADT).*

Resumo. *Este relatório apresenta uma lista duplamente encadeada escrita na linguagem "c" de programacção, tal como sua implementacção. O programa foi implementado utilizando tipo abstrato de dados (TAD).*

1. Introducção

Uma lista duplamente encadeada segue os mesmos princípios de uma lista simplesmente encadeada, porém com cada nó presenta na lista contendo não só um ponteiro que aponta para o próximo nó da lista, como também um ponteiro indicando qual o nó anterior de cada item. A lista estudada é composta por 13 funcções que serão descritas nos seguintes tópicos deste relatório:

1. criarValor
2. alocarCelula
3. alocarLista
4. liberarLista
5. inserirInicioLista
6. inserirFinalLista
7. inserirOrdemLista
8. removerInicioLista
9. removerFinalLista
10. removerEspecifico
11. buscaCelulaPosicao
12. buscaCelulaDado
13. imprimirLista

2. criarValor

Essa funcção não recebe nenhum parâmetro. Dentro dela é declarado um valor inteiro que será retornado. Esse "valor" será utilizado na criacção de nós e na remocção específica de um nó.

3. alocarCelula

Essa funcção não recebe nenhum parâmetro. Quando chamada, a funcção aloca memória equivalente a uma célula (nó) e a retorna.

4. alocarLista

Essa função não recebe nenhum parâmetro. Quando chamada, a função aloca memória equivalente a uma lista, que é um ponteiro que aponta para um nó, servindo como referência para o início da lista, e o retorna.

5. liberarLista

Essa função retorna um valor inteiro e recebe uma lista (ponteiro que aponta para um nó) como parâmetro. Em seguida é feita a verificação da existência da lista referente. Caso a lista não exista, ou seja, não tenha sido alocada ou houve algum erro na alocação, a função se encerra retornando o valor 0. Caso a lista exista, um nó auxiliar será criado para que seja possível liberar cada nó presente na lista referente. Esse passo ocorre por meio de um loop que percorre os nós até um último, verificando se o ponteiro que aponta para o próximo item de cada nó é nulo, indicando a chegada no último item da lista e liberando o mesmo. Esse passo é repetido até que não haja mais nós na lista. Por fim o ponteiro que indica o início da lista também é liberado e a função retorna o valor 1.

6. inserirInicioLista

Essa função retorna um valor inteiro e recebe uma lista (ponteiro que aponta para um nó) como parâmetro. Em seguida é feita a verificação da existência da lista referente. Caso a lista não exista, ou seja, não tenha sido alocada ou houve algum erro na alocação, a função se encerra retornando o valor 0. Caso a lista exista, um valor inteiro e um nó serão criados. Se o ponteiro de inicio da lista for nulo, ou seja, a lista não contém nenhum nó, ele simplesmente passará apontar para o nó recém criado. Caso a lista não esteja vazia, o ponteiro anterior do novo nó apontará para um valor nulo, o ponteiro posterior receberá o endereço de onde aponta a lista e o ponteiro da lista receberá o endereço do nó recém criado. O valor 1 é retornado.

7. inserirFinalLista

Essa função retorna um valor inteiro e recebe uma lista (ponteiro que aponta para um nó) como parâmetro. Em seguida é feita a verificação da existência da lista referente. Caso a lista não exista, ou seja, não tenha sido alocada ou houve algum erro na alocação, a função se encerra retornando o valor 0. Caso a lista exista, um valor inteiro e um nó serão criados. Se o ponteiro de inicio da lista for nulo, ou seja, a lista não contém nenhum nó, ele simplesmente passará apontar para o nó recém criado. Caso a lista não esteja vazia, um loop se encarregará de percorrer a lista em busca do último nó, o qual o ponteiro posterior aponta um valor nulo. Esse nó passa a apontar seu ponteiro posterior para o nó recém criado, o nó recém criado por sua vez aponta seu ponteiro anterior para o atual último nó da lista e seu ponteiro posterior para um valor nulo, se tornando assim o último item da lista. O valor 1 é retornado.

8. inserirOrdemLista

Essa função retorna um valor inteiro e recebe uma lista (ponteiro que aponta para um nó) como parâmetro. Em seguida é feita a verificação da existência da lista referente. Caso a lista não exista, ou seja, não tenha sido alocada ou houve algum erro na alocação a função se encerra retornando o valor 0. Caso a lista exista, um valor inteiro e um

nó serão criados. Se o ponteiro de inicio da lista for nulo, ou seja, a lista não contém nenhum nó, ele simplesmente passará apontar para o nó recém criado. Caso a lista não esteja vazia, um loop se encarregará de percorrer a lista em busca do nó que carrega o valor imediatamente menor ao valor recebido pela variável "valor" da função, que indica a posição de inserção do novo nó. Assim que encontrado, o ponteiro posterior do novo nó aponta para o mesmo endereço do ponteiro posterior do nó encontrado, o ponteiro anterior do nó seguinte aponta para o nó novo, o ponteiro anterior do nó novo aponta para o nó encontrado e o por fim, o ponteiro posterior do nó encontrado aponta para o nó novo. O valor 1 é retornado.

9. removerInicioLista

Essa função retorna um valor inteiro e recebe uma lista (ponteiro que aponta para um nó) como parâmetro. Em seguida é feita a verificação da existência da lista referente. Caso a lista não exista, ou seja, não tenha sido alocada ou houve algum erro na alocação, a função se encerra retornando o valor 0. Em seguida é verificado se a lista está ou não vazia, encerrando a função e retornando valor 0 caso esteja. Caso a lista exista e não esteja vazia, um nó será criado. Esse nó recebe o endereço apontado pelo ponteiro de inicio da lista. Em seguida o ponteiro anterior do nó posterior passará apontar um valor nulo e o nó recém criado será liberado. O valor 1 é retornado.

10. removerFinalLista

Essa função retorna um valor inteiro e recebe uma lista (ponteiro que aponta para um nó) como parâmetro. Em seguida é feita a verificação da existência da lista referente. Caso a lista não exista, ou seja, não tenha sido alocada ou houve algum erro na alocação, a função se encerra retornando o valor 0. Em seguida é verificado se a lista está ou não vazia, encerrando a função e retornando valor 0 caso esteja. Caso a lista exista e não esteja vazia, um nó será criado. Um loop se encarregará de percorrer os nós da lista até que seja encontrado um nó cujo o ponteiro posterior aponte um valor nulo. O nó criado receberá o endereço do nó encontrado, o ponteiro posterior do nó anterior ao encontrado apontará para um valor nulo e o nó criado será liberado. O valor 1 é retornado.

11. removerEspecifico

Essa função retorna um valor inteiro e recebe uma lista (ponteiro que aponta para um nó) como parâmetro. Em seguida é feita a verificação da existência da lista referente. Caso a lista não exista, ou seja, não tenha sido alocada ou houve algum erro na alocação, a função se encerra retornando o valor 0. Em seguida é verificado se a lista está ou não vazia, encerrando a função e retornando valor 0 caso esteja. Caso a lista exista e não esteja vazia, um nó e um valor serão criados. Um loop se encarregará de percorrer os nós da lista até que seja encontrado um nó cujo o valor seja equivalente ao valor criado. O nó criado receberá o endereço do nó encontrado, o ponteiro posterior do nó anterior ao encontrado apontará para o nó posterior ao encontrado e o ponteiro anterior do nó posterior ao encontrado apontará para o nó anterior ao encontrado. O nó criado será liberado. O valor 1 é retornado.

12. buscaCelulaPosicao

Essa função retorna um valor inteiro e recebe uma lista (ponteiro que aponta para um nó) como parâmetro. Em seguida é feita a verificação da existência da lista referente. Caso

a lista não exista, ou seja, não tenha sido alocada ou houve algum erro na alocação a função se encerra retornando o valor 0. Em seguida é verificado se a lista está ou não vazia, encerrando a função e retornando valor 0 caso esteja. Caso a lista exista e não esteja vazia, um nó e um valor serão criados. Um loop se encarregará de percorrer os nós da lista até que seja encontrado um nó cujo o valor seja equivalente ao valor criado. O nó criado receberá o endereço do nó encontrado, o ponteiro posterior do nó anterior ao encontrado apontará para o nó posterior ao encontrado e o ponteiro anterior do nó posterior ao encontrado apontará para o nó anterior ao encontrado. O nó criado será liberado. O valor 1 é retornado.

13. buscaCelulaDado

Essa função retorna um valor inteiro e recebe uma lista (ponteiro que aponta para um nó) como parâmetro. Em seguida é feita a verificação da existência da lista referente. Caso a lista não exista, ou seja, não tenha sido alocada ou houve algum erro na alocação a função se encerra retornando o valor 0. Em seguida é verificado se a lista está ou não vazia, encerrando a função e retornando valor 0 caso esteja. Caso a lista exista e não esteja vazia, um nó e um valor serão criados. Um loop se encarregará de percorrer os nós da lista até que seja encontrado um nó cujo o valor seja equivalente ao valor criado. O nó criado receberá o endereço do nó encontrado, o ponteiro posterior do nó anterior ao encontrado apontará para o nó posterior ao encontrado e o ponteiro anterior do nó posterior ao encontrado apontará para o nó anterior ao encontrado. O nó criado será liberado. O valor 1 é retornado.

14. imprimirLista

Essa função retorna um valor inteiro e recebe uma lista (ponteiro que aponta para um nó) como parâmetro. Em seguida é feita a verificação da existência da lista referente. Caso a lista não exista, ou seja, não tenha sido alocada ou houve algum erro na alocação a função se encerra retornando o valor 0. Caso a lista exista, um nó será criado. Esse nó recebe o endereço do primeiro nó da lista e um loop se encarrega de percorrer os nós até que um deles aponte posteriormente para um valor nulo imprimindo cada um deles. O valor 1 é retornado.

Referências

https://www.youtube.com/playlist?list=PL8iN9FQ7_jt5PigiANscK2ZXqPJJ52DAT