INF1010 - ESTRUTURAS DE DADOS AVANCADAS - 2022.1 - 3WB

Laboratório 2 de Estruturas de Dados

Lista Duplamente Encadeada

Eric Ruas Leão - 2110694
Gusthavo Macedo - 2021030
Marina Schuler Martins - 2110075
Wladimir Ramos - 2110104

Lista Duplamente Encadeada

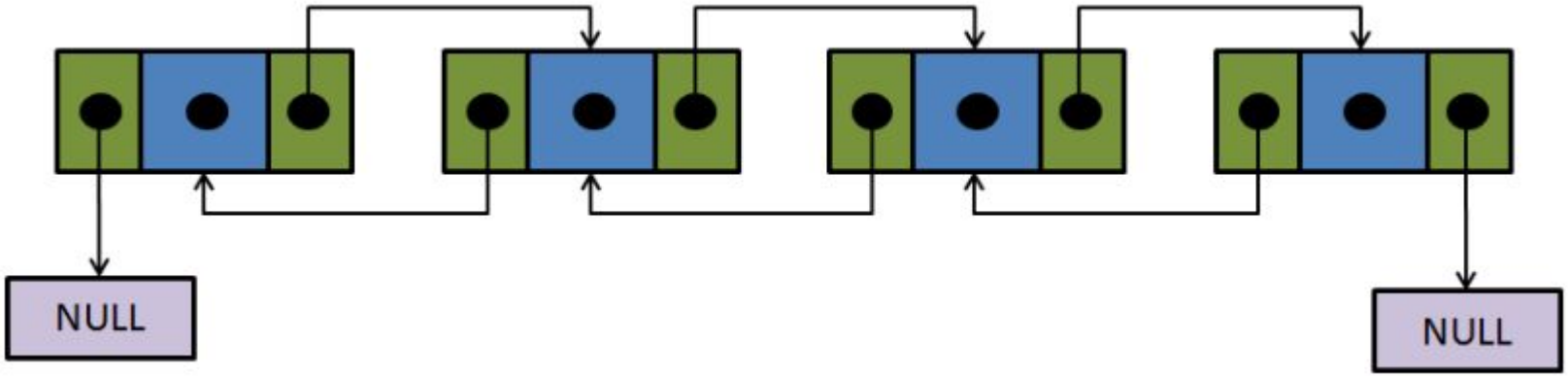
Na lista duplamente encadeada, cada elemento tem um ponteiro para o próximo elemento e outro para o elemento anterior. Portanto, dado um elemento qualquer, conseguimos acessar o próximo e o anterior, e dado um ponteiro para o último elemento da lista, conseguimos percorrer a lista em ordem inversa.

```c
#include <stdio.h>
#include <stdlib.h>

/*estrutura da lista*/
struct lista {
  int info;
  struct lista* ant;
  struct lista* prox;
};

typedef struct lista Lista;
```

Lista Duplamente Encadeada

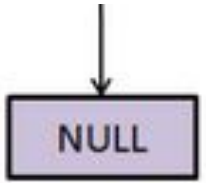


Funções Implementadas

```
Lista *busca(Lista * lst,int val);
Lista *criaLst(void);
void liberaLst(Lista *lst);
Lista* lst_insere (Lista* lst, int val);
Lista* lst_retira (Lista* lst, int val);
int lst_vazia (Lista* lst);
void imprime_lista(Lista *lst);
```

Cria Lista

```c
/*cria lista*/
Lista *criaLst(void){
  return NULL;
}
```

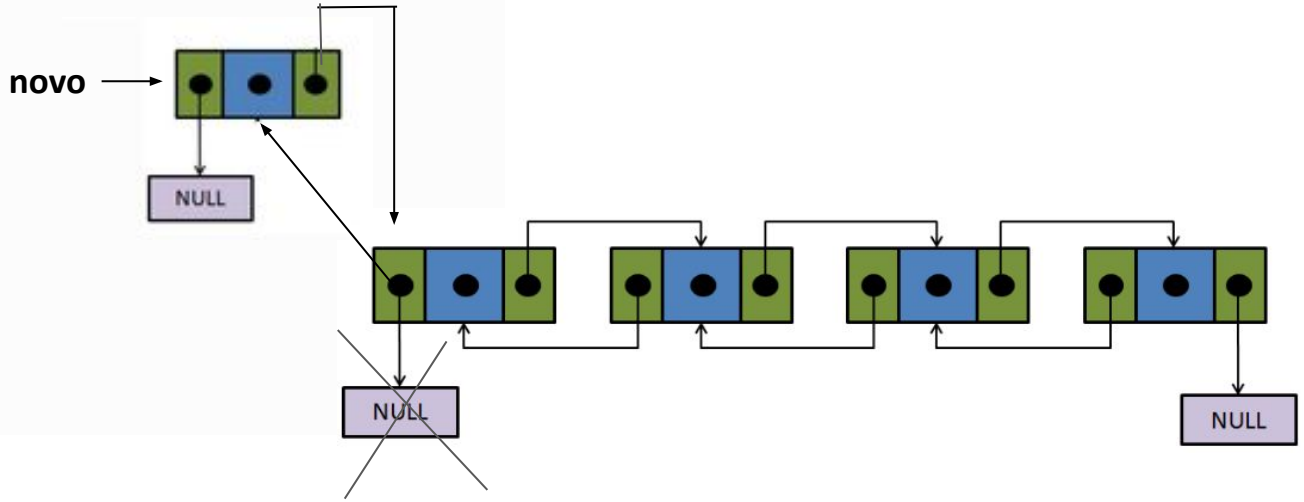


Libera Lista

```c
/*libera lista*/
void liberaLst(Lista *lst){
  Lista *topo = lst;
  while (topo != NULL) {
    Lista *proximo = topo->prox;
    free(topo);
    topo = proximo;
  }
}
```

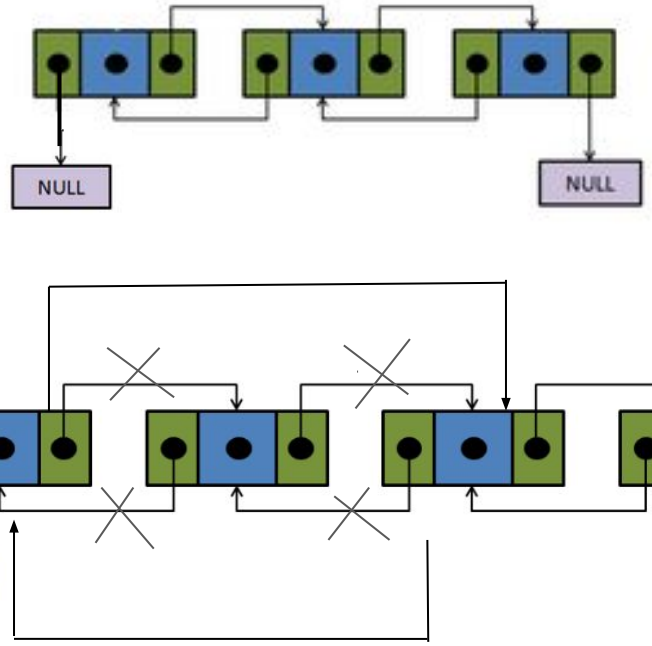
Insere na Lista

```c
/*insere na lista*/
Lista* lst_insere (Lista* lst, int val){
  Lista* novo = (Lista*) malloc(sizeof(Lista));
  if (novo == NULL){
    printf("Nao foi possivel alocar espaco.\n");
    exit(1);
  }
  novo->info = val;
  novo->prox = lst;
  novo->ant = NULL;
  if (lst != NULL){
    lst->ant = novo;
  }
  return novo;
}
```



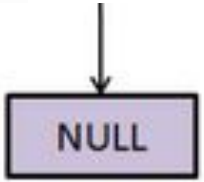
Retira da Lista

```c
/*retira da lista*/
Lista* lst_retira (Lista* lst, int val){
  Lista* p = busca(lst,val);
  if (p == NULL){
    return lst;
  }if (lst == p){
    lst = p->prox;
  }else{
    p->ant->prox = p->prox;
  }if (p->prox != NULL) {
    p->prox->ant = p->ant;
  }
  free(p);
  return lst;
}
```



Verifica se a Lista está Vazia

```c
/*verifica se lista está vazia*/
int lst_vazia (Lista* lst) {
  return (lst == NULL);
}
```



Exibe a Lista

```c
/*exibe a lista*/
void imprime_lista(Lista *lst){
  Lista *p;
  for (p = lst;p != NULL;p = p->prox)
    printf("%d,",p->info);
  return;
}
```

Busca na Lista

```c
/*busca na lista*/
Lista *busca(Lista * lst,int val){
  Lista * aux;
  for ( aux = lst; aux != NULL; aux = aux->prox)
    if(aux->info == val)
      return aux;
  return NULL;
}
```