

File: lista00.c

```
#include <stdio.h>
#include <stdlib.h>
#include "lista00.h"
struct lista
{
    // Criamos estrutura do nó
    int info;
    struct lista* ptrProx; //Ponteiro para o próximo nó da lista
};
//Vamos a criar uma função que vai criar uma lista
//Criamos uma lista vazia
Lista01* criarLista (){
    return NULL;
}

/*A função recebe dos parâmetros de entrada, uma lista sobre
a qual será inserido o novo elemento, como também a informação
do novo elemento.
Ela tem como valor retorno a nova lista */
Lista01* inserirLista(Lista01* argLista, int argInfo){
    Lista01* novo =(Lista01* )malloc(sizeof(Lista01));
    novo-> info =argInfo;
    novo-> ptrProx =argLista;
    return novo;
}
//imprimir os elementos da lista
void imprimirLista(Lista01* argLista){
    Lista01* ptrTrb;
    for(ptrTrb=argLista; ptrTrb != NULL; ptrTrb= ptrTrb -> ptrProx )
        printf("%d\n", ptrTrb -> info);
}
//Função para verificar se a lista está vazia
int testarVazia(Lista01* argLista){
    if(argLista == NULL)
        return 1;
    return 0;
}

//Buscar Conteúdo nos elementos da lista
Lista01* busca(Lista01* argLista, int infoBusca){
    Lista01* ptrTrb;
    for(ptrTrb=argLista; ptrTrb!= NULL; ptrTrb=ptrTrb->ptrProx ){
        if(ptrTrb-> info == infoBusca)
            return ptrTrb;
    }
    return NULL;
}

//Função para excluir elementos da lista
Lista01* excluirLista(Lista01* argLista, int infoBusca){
    Lista01* ptrAnt=NULL;
    Lista01* ptrTrb =argLista;
    while (ptrTrb!=NULL && ptrTrb-> info != infoBusca ){
        ptrAnt = ptrTrb;
        ptrTrb = ptrTrb->ptrProx;
    }
    if (ptrTrb == NULL)
        return argLista;
    if (ptrAnt == NULL){
        //remove do IndIniLst da lista
        argLista = ptrTrb -> ptrProx;
    }
    else
    {
        ptrAnt-> ptrProx = ptrTrb-> ptrProx;
    }
}
```

```

        }
        free(ptrTrb);
        return argLista;
    }

//Função para Liberar a lista
void liberar(Lista01* argLista){
    Lista01* ptrTrb;
    while(ptrTrb !=NULL ){
        Lista01* ptrTemp = ptrTrb-> ptrProx;
        free(ptrTrb);
        ptrTrb=ptrTemp;
    }
}

```

File: lista00.h

```

typedef struct lista Lista01; // Representa um nó da lista
//Função de criação de lista
Lista01* criarLista();

//Função que inseri no inicio da lista

/*A função recebe dos parâmetros de entrada, uma lista sobre
a qual será inserido o novo elemento, como também a informação
do novo elemento.
Ela tem como valor retorno a nova lista */

Lista01* inserirLista(Lista01* argLista, int argInfo);

//Função para imprimir os elementos da lista
void imprimirLista(Lista01* argLista);

//Função para verificar se a lista está vazia
int testarVazia(Lista01* argLista);

//Buscar elementos na lista
Lista01* busca(Lista01* argLista, int infoBusca);

//Função para excluir elementos da lista
Lista01* excluirLista(Lista01* argLista, int infoBusca);

//Função para Liberar a lista
void liberar(Lista01* argLista);

//Inserir no Inicio da Lista
//Lista01* insereInicio(Lista01* argLista, int valor, t_lista * l )

```

File: main.c

```

#include <stdio.h>
#include "lista00.h"

int main(int argc, char* argv[])
{
    Lista01* li;
    li = criarLista();
    li = inserirLista(li, 10);
    li = inserirLista(li, 20);
    li = inserirLista(li, 30);
    li = inserirLista(li, 40);
    li = inserirLista(li, 50);
    imprimirLista(li);
    if(testarVazia(li))
        printf("Lista Vazia !\n");
    else

```

```
        printf("Lista Nao Vazia \n");

int elemento = 100;
if( busca(li, elemento) == NULL)
    printf("Não encontrou o elemento %d\n", elemento);
else
    printf("Encontrou o elemento %d\n", elemento);

li = excluirLista(li,10);
    imprimirLista(li);
liberar(li);
li = inserirLista(li, 50);

return 0;
}
```