

Nomes:

Henrique Sartori Siqueira

19240472

Rafael Silva Barbon

19243633

EX1:

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

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

```
/*Faça um programa que receba uma lista de inteiros e remova todos os  
valores  
repetidos desta lista.  
*/
```

```
typedef struct lista{  
    int info;  
    struct lista *prox;  
}Lista;
```

```
Lista * InicializaLista(){  
    return NULL;  
}
```

```
void InsereLista(Lista **recebida, int valor){  
    Lista *novo = (Lista *)malloc(sizeof(Lista));  
  
    novo->info = valor;  
    novo->prox = (*recebida);  
    (*recebida) = novo;  
}
```

```
void ImprimeLista(Lista *recebida){  
    Lista *aux = recebida;  
  
    printf("\n\tLista: ");  
    while(aux != NULL){
```

```

        printf("\n\t%d",aux->info);
        aux = aux->prox;
    }
}

void Repetidos(Lista *recebida){
    Lista *aux = recebida, *aux2act, *aux2prev;
    int var;

    for(; aux != NULL; aux = aux->prox){
        var = aux->info;
        aux2act = aux->prox;
        aux2prev = aux;
        while(aux2act != NULL){
            if(aux2act->info == var){
                aux2prev->prox = aux2act->prox;
                free(aux2act);
                aux2act = aux2prev->prox;
            }else{
                aux2prev = aux2act;
                aux2act = aux2prev->prox;
            }
        }
    }
}

```

```

void DesalocaLista(Lista *recebida){
    Lista *aux = recebida, *aux2;

    while(aux != NULL){
        aux2 = aux->prox;
        free(aux);
        aux = aux2;
    }
}

```

```

int main(){
    Lista *l = InicializaLista();
    int v;

    do{
        printf("\n\tInsira um valor para a lista (-555 para continuar):
");
        scanf("%d",&v);
        if(v != -555)
            InsereLista(&l,v);
    }while(v != -555);
    ImprimeLista(l);

    Repetidos(l);

    ImprimeLista(l);

    DesalocaLista(l);
    return 0;
}

```

EX2:

```

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

```

```

/*Faça um programa que receba uma lista de inteiros e remova todos os
valores
repetidos desta lista.
*/

```

```

typedef struct lista{
    int info;
    struct lista *prox;
}Lista;

```

```
Lista * InicializaLista(){
    return NULL;
}
```

```
void InsereLista(Lista **recebida, int valor){
    Lista *novo = (Lista *)malloc(sizeof(Lista));

    novo->info = valor;
    novo->prox = (*recebida);
    (*recebida) = novo;
}
```

```
void InserePos(Lista **L3,int v, int pos){
    Lista *novo = (Lista *)malloc(sizeof(Lista)), *aux = (*L3)->prox,
    *prev = (*L3);
    int i = 1;

    novo->info = v;
    while(aux != NULL){
        if(i == pos){
            novo->prox = aux;
            prev->prox = novo;
            i = -1;
            break;
        }
        prev = aux;
        aux = aux->prox;
        i++;
    }
    if(i != -1){
        novo->prox = aux;
        prev->prox = novo;
    }
}
```

```

void ImprimeLista(Lista *recebida){
    Lista *aux = recebida;

    printf("\n\tLista: ");
    while(aux != NULL){
        printf("\n\t%d",aux->info);
        aux = aux->prox;
    }
}

```

```

Lista * juntalistas(Lista *L1, Lista *L2){
    Lista *L3 = InicializaLista(), *aux = L1->prox, *aux2;
    int v = L1->info, cont;

```

InsereLista(&L3,v); //insere um elemento na lista para poder entrar
 no loop

```

        while(aux != NULL){
            v = aux->info;
            aux2 = L3;
            cont = 0; //contador da posicao que inserira na lista
            while(aux2 != NULL){
                if(aux2->info > v)
                    cont++;
            }
            else{
                cont != 0 ? InserePos(&L3,v,cont) : InsereLista(&L3,v); // se
for = 0 -> primeiro da lista, senao no meio
                cont = -1;
                break;
            }
            aux2 = aux2->prox;
        }
        if(cont != -1)//inserir no final da lista
            InserePos(&L3,v,cont);
        aux = aux->prox;

```

```

    }
    aux = L2;
    while(aux != NULL){
        v = aux->info;
        aux2 = L3;
        cont = 0; //contador da posicao que inserira na lista
        while(aux2 != NULL){
            if(aux2->info > v)
                cont++;
            else{
                cont != 0 ? InserePos(&L3,v,cont) :
InsereLista(&L3,v); // se for != 0 meio da lista, senao no inicio
                cont = -1;
                break;
            }
            aux2 = aux2->prox;
        }
        if(cont != -1)//inserir no final da lista
            InserePos(&L3,v,cont);
        aux = aux->prox;
    }

    return L3;
}

```

```

void DesalocaLista(Lista *recebida){
    Lista *aux = recebida, *aux2;

    while(aux != NULL){
        aux2 = aux->prox;
        free(aux);
        aux = aux2;
    }
}

```

```

int main(){
    Lista *L1 = InicializaLista(), *L2 = InicializaLista(), *L3;
    int v;

    do{
        printf("\n\tInsira um valor para a lista 1 (-555 para continuar):
");
        scanf("%d",&v);
        if(v != -555)
            InsereLista(&L1,v);
    }while(v != -555);
    do{
        printf("\n\tInsira um valor para a lista 2 (-555 para continuar):
");
        scanf("%d",&v);
        if(v != -555)
            InsereLista(&L2,v);
    }while(v != -555);

    ImprimeLista(L1);
    ImprimeLista(L2);

    L3 = juntalistas(L1,L2);

    ImprimeLista(L3);

    DesalocaLista(L1);
    DesalocaLista(L2);
    DesalocaLista(L3);
    return 0;
}

```

EX3:

```

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

```

/*Faça um programa que receba e armazene diversos valores em uma lista de inteiros. O usuário poderá buscar por um elemento na lista e, sempre que ele fizer isso, o elemento que ele buscou deverá ser movido para o primeiro lugar da lista (isso irá fazer com que os elementos mais buscados sejam encontrados mais rapidamente)

*/

```
struct lista
{
    int info;
    struct lista *prox;
};
typedef struct lista Lista;
```

```
Lista* CriaLista(void)
{
    return NULL;
}
```

```
Lista* Insere(Lista *recebida, int valor)
{
    Lista *novo=(Lista*)malloc(sizeof(Lista));
    novo->info=valor;
    novo->prox=recebida;
    return novo;
}
```

```
int VaziaLista(Lista *recebida)
{
    if(recebida == NULL)
        return 1;
    return 0;
}
```

```
void Imprime(Lista *recebida)
```



```

{
    Lista *aux;
    if(VaziaLista(recebida))
        printf("\nLista Vazia!");
    for(aux = recebida; aux!= NULL; aux = aux->prox)
        printf("\nInfo: %d", aux->info);
}

Lista* busca_e_move(Lista *recebida, int valor)
{
    Lista *aux=recebida, *anterior=NULL;
    int flag=0;
    while(!VaziaLista(aux))
    {
        if(aux->info==valor)
        {
            printf("\nElemento encontrado");
            flag=1;
            if(anterior==NULL)//Já é o primeiro elemento
                break;
            else
            {
                anterior->prox=aux->prox;
                aux->prox=recebida;
                return aux;
            }
        }
        anterior = aux;
        aux = aux->prox;
    }
    if(!flag)
        printf("\nElemento nao encontrado");
    return recebida;
}

```

```

Lista* libera(Lista *recebida)
{
    Lista *aux;
    while(!VaziaLista(recebida))
    {
        aux=recebida->prox;
        free(recebida);
        recebida=aux;
    }
    return NULL;
}

```

```

int main()
{
    Lista *L=CriaLista();
    int info;
    do{
        printf("\nInsira elementos na Lista(-999 p/ sair:");
        scanf("%d",&info);
        if(info != -999)
            L=Insere(L,info);
    }while(info != -999);
    system("clear");

    do{
        printf("\nLista Lida:");
        Imprime(L);
        printf("\nDigite o elemento que deseja buscar na lista(-999 p/
sair:");
        scanf("%d", &info);
        if(info != -999)
            L=busca_e_move(L, info);
    }while(info != -999);
    L=libera(L);
    if(VaziaLista(L))

```

```
        printf("\nEncerrado com sucesso!");  
    return 0;  
}
```