**Programa principal**

#include <stdio.h>

#include <stdlib.h>

#include "ListaSequencial.h"

int main(){

struct aluno a[4] = {{2,"Andre",9.5,7.8,8.5},

{4,"Ricardo",7.5,8.7,6.8},

{1,"Bianca",9.7,6.7,8.4},

{3,"Ana",5.7,6.1,7.4}};

Lista\* li = cria\_lista();

int i;

for(i=0; i < 4; i++)

insere\_lista\_ordenada(li,a[i]);

imprime\_lista(li);

printf("\n\n\n\n");

for(i=0; i < 5; i++){

if (!remove\_lista\_otimizado(li,i))

printf("Erro\n");

imprime\_lista(li);

printf("\n\n\n\n");

}

libera\_lista(li);

system("pause");

return 0;

}

**ListaSequencial.h**

//Arquivo ListaSequencial.h

#define MAX 100

struct aluno{

int matricula;

char nome[30];

float n1,n2,n3;

};

typedef struct lista Lista;

Lista\* cria\_lista();

void libera\_lista(Lista\* li);

int consulta\_lista\_pos(Lista\* li, int pos, struct aluno \*al);

int consulta\_lista\_mat(Lista\* li, int mat, struct aluno \*al);

int insere\_lista\_final(Lista\* li, struct aluno al);

int insere\_lista\_inicio(Lista\* li, struct aluno al);

int insere\_lista\_ordenada(Lista\* li, struct aluno al);

int remove\_lista(Lista\* li, int mat);

int remove\_lista\_inicio(Lista\* li);

int remove\_lista\_final(Lista\* li);

int tamanho\_lista(Lista\* li);

int lista\_cheia(Lista\* li);

int lista\_vazia(Lista\* li);

void imprime\_lista(Lista\* li);

int remove\_lista\_otimizado(Lista\* li, int mat);

**ListaSequencial.h**

#include <stdio.h>

#include <stdlib.h>

#include "ListaSequencial.h" //inclui os Protótipos

//Definição do tipo lista

struct lista{

int qtd;

struct aluno dados[MAX];

};

Lista\* cria\_lista(){

Lista \*li;

li = (Lista\*) malloc(sizeof(struct lista));

if(li != NULL)

li->qtd = 0;

return li;

}

void libera\_lista(Lista\* li){

free(li);

}

int consulta\_lista\_pos(Lista\* li, int pos, struct aluno \*al){

if(li == NULL || pos <= 0 || pos > li->qtd)

return 0;

\*al = li->dados[pos-1];

return 1;

}

int consulta\_lista\_mat(Lista\* li, int mat, struct aluno \*al){

if(li == NULL)

return 0;

int i = 0;

while(i<li->qtd && li->dados[i].matricula != mat)

i++;

if(i == li->qtd)//elemento nao encontrado

return 0;

\*al = li->dados[i];

return 1;

}

int insere\_lista\_final(Lista\* li, struct aluno al){

if(li == NULL)

return 0;

if(li->qtd == MAX)//lista cheia

return 0;

li->dados[li->qtd] = al;

li->qtd++;

return 1;

}

int insere\_lista\_inicio(Lista\* li, struct aluno al){

if(li == NULL)

return 0;

if(li->qtd == MAX)//lista cheia

return 0;

int i;

for(i=li->qtd-1; i>=0; i--)

li->dados[i+1] = li->dados[i];

li->dados[0] = al;

li->qtd++;

return 1;

}

int insere\_lista\_ordenada(Lista\* li, struct aluno al){

if(li == NULL)

return 0;

if(li->qtd == MAX)//lista cheia

return 0;

int k,i = 0;

while(i<li->qtd && li->dados[i].matricula < al.matricula)

i++;

for(k=li->qtd-1; k >= i; k--)

li->dados[k+1] = li->dados[k];

li->dados[i] = al;

li->qtd++;

return 1;

}

int remove\_lista(Lista\* li, int mat){

if(li == NULL)

return 0;

if(li->qtd == 0)

return 0;

int k,i = 0;

while(i<li->qtd && li->dados[i].matricula != mat)

i++;

if(i == li->qtd)//elemento nao encontrado

return 0;

for(k=i; k< li->qtd-1; k++)

li->dados[k] = li->dados[k+1];

li->qtd--;

return 1;

}

int remove\_lista\_otimizado(Lista\* li, int mat){

if(li == NULL)

return 0;

if(li->qtd == 0)

return 0;

int i = 0;

while(i<li->qtd && li->dados[i].matricula != mat)

i++;

if(i == li->qtd)//elemento nao encontrado

return 0;

li->qtd--;

li->dados[i] = li->dados[li->qtd];

return 1;

}

int remove\_lista\_inicio(Lista\* li){

if(li == NULL)

return 0;

if(li->qtd == 0)

return 0;

int k = 0;

for(k=0; k< li->qtd-1; k++)

li->dados[k] = li->dados[k+1];

li->qtd--;

return 1;

}

int remove\_lista\_final(Lista\* li){

if(li == NULL)

return 0;

if(li->qtd == 0)

return 0;

li->qtd--;

return 1;

}

int tamanho\_lista(Lista\* li){

if(li == NULL)

return -1;

else

return li->qtd;

}

int lista\_cheia(Lista\* li){

if(li == NULL)

return -1;

return (li->qtd == MAX);

}

int lista\_vazia(Lista\* li){

if(li == NULL)

return -1;

return (li->qtd == 0);

}

void imprime\_lista(Lista\* li){

if(li == NULL)

return;

int i;

for(i=0; i< li->qtd; i++){

printf("Matricula: %d\n",li->dados[i].matricula);

printf("Nome: %s\n",li->dados[i].nome);

printf("Notas: %f %f %f\n",li->dados[i].n1,

li->dados[i].n2,

li->dados[i].n3);

printf("-------------------------------\n");

}

}