Aluno: Gustavo Cesar Silveira Pontual

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

typedef struct contato{

char nome[100];

char fone[15];

struct contato \*ant, \*prox;

}TContato;

typedef struct indice{

char letra;

struct indice \*ant,\*prox;

TContato \*inicio;

}TIndice;

void cadastrar(TIndice \*\*Li,char nome[]);

void exibir\_fone(TIndice \*L);

void alterar\_fone(TIndice \*L);

void exibir\_agenda(TIndice \*L);

void remover\_contato(TIndice \*\*L);

void limpar\_lista(TIndice \*\*L);

TIndice \*verificar\_indice(TIndice \*L,char n[]);

TIndice \*verificar\_indice\_posi(TIndice \*L,char n[]);

TContato \*verificar\_contato\_posi(TContato \*L,char n[]);

TContato \*busca\_contato\_nome\_posi(TIndice \*Li,char n[]);

TContato \*verificar\_contato(TContato \*L,char n[]);

int main(){

TContato \*BuscaC;

TIndice \*Lista\_indice=NULL,\*aux;

char op=' ',nome\_aux[100],op2;

do{

if(op>=48 &&op<=57){

system("pause");

system("cls");

}

printf(" M E N U\n");

printf("1-Cadastrar\n2-Remover\n3-Exibir telefone do contato\n4-Alterar telefone\n5-Exibir Agenda\n"

"6-Apagar agenda\n7-Sair\nDigite a opcao: ");

scanf("%c",&op);fflush(stdin);

switch(op){

case '1':

printf("\nDigite o nome ");

do{

gets(nome\_aux);fflush(stdin);

BuscaC = busca\_contato\_nome\_posi(Lista\_indice,nome\_aux);

if(BuscaC!=NULL)

printf("Nome existente ! Digite outro nome: ");

}while(BuscaC!=NULL);

cadastrar(&Lista\_indice,nome\_aux);

break;

case '2':

remover\_contato(&Lista\_indice);

break;

case '3':

exibir\_fone(Lista\_indice);

break;

case '4':

alterar\_fone(Lista\_indice);

break;

case '5':

exibir\_agenda(Lista\_indice);

break;

case '6':

limpar\_lista(&Lista\_indice);

break;

case '7':

break;

default:

printf("Opcao invalida !");

break;

}

}while(op!='7');

return 0;

}

void cadastrar(TIndice \*\*Li,char nome[]){

TContato \*novo\_c,\*aux\_c,\*pos\_c;

TIndice \*novo\_i,\*aux\_i=\*Li,\*pos\_i;

if(\*Li==NULL){// se a lista de Indices(Letras) esta vazia , adiciona

novo\_i = (TIndice \*) malloc(sizeof(TIndice));

\*Li = novo\_i;

novo\_i ->letra = toupper(nome[0]);

novo\_i->prox=novo\_i;

novo\_i->ant=novo\_i;

novo\_i->inicio=NULL;

}

else{

aux\_i=verificar\_indice(\*Li,nome);// vai retornar o endereço do indice, se existe algum indice com o nome dado

if(aux\_i==NULL){// se n existir , cria um indice

pos\_i=verificar\_indice\_posi(\*Li,nome);//retorna a posisao do indice(letra) q seja maior q o novo indice

novo\_i=(TIndice \*) malloc(sizeof(TIndice));

novo\_i->letra = nome[0];

if(pos\_i->letra < novo\_i->letra){// adiciona no fim

novo\_i->prox=pos\_i->prox;

novo\_i->ant=pos\_i;

pos\_i->prox->ant=novo\_i;

pos\_i->prox=novo\_i;

novo\_i->inicio=NULL;

}

else{// adiciona no começo e no meio

novo\_i->prox=pos\_i;

novo\_i->ant=pos\_i->ant;

pos\_i->ant->prox=novo\_i;

pos\_i->ant=novo\_i;

novo\_i->inicio=NULL;

}

}

}

aux\_i=verificar\_indice(\*Li,nome);

novo\_c = (TContato \*)malloc(sizeof(TContato));

strcpy(novo\_c->nome,nome);fflush(stdin);

printf("Digite o tefone : ");

gets(novo\_c->fone);fflush(stdin);

if(novo\_c->nome[0]>=97 && novo\_c->nome[0]<=122)

novo\_c->nome[0]=toupper(novo\_c->nome[0]);

if(aux\_i->inicio==NULL){//se n existir contato com a letra desde indice

novo\_c->prox = novo\_c;

novo\_c->ant=novo\_c;

aux\_i->inicio=novo\_c;

}

else{

nome = novo\_c->nome;

pos\_c=verificar\_contato\_posi(aux\_i->inicio,nome);// retorna a posisao do contato com ordem alfabetica maior q a do nome q sera cadastrado

if((pos\_c->prox == aux\_i->inicio) && strcmp(nome,pos\_c->nome)>0){//add no fim

novo\_c->prox=pos\_c->prox;

novo\_c->ant=pos\_c;

novo\_c->prox->ant=novo\_c;

pos\_c->prox=novo\_c;

}

else if(pos\_c->ant->prox == aux\_i->inicio && strcmp(nome,pos\_c->nome)<0){// add inicio

novo\_c->prox=pos\_c;

novo\_c->ant=pos\_c->ant;

pos\_c->ant->prox=novo\_c;

pos\_c->ant=novo\_c;

aux\_i->inicio = novo\_c;

}

else{//add meio

novo\_c->prox=pos\_c;

novo\_c->ant=pos\_c->ant;

pos\_c->ant->prox=novo\_c;

pos\_c->ant=novo\_c;

}

}

}

TIndice \*verificar\_indice(TIndice \*L,char nome[]){

TIndice \*aux=L;

char letra;

if(L==NULL)

printf("Lista Vazia!\n");

else{

letra=toupper(nome[0]);

do{

if(aux->letra==letra)

return aux;

aux=aux->prox;

}while(aux!=L);

}

return NULL;

}

TIndice \*verificar\_indice\_posi(TIndice \*L,char n[]){

TIndice \*aux=L;

char letra;

if(L==NULL){

printf("Lista Vazia!\n");

return NULL;

}

else{

letra=toupper(n[0]);

do{

if(aux->letra > letra)

return aux;

aux=aux->prox;

}while(aux!=L);

return aux->ant;

}

}

TContato \*verificar\_contato\_posi(TContato \*L,char n[]){

TContato \*aux=L;

if(L==NULL){

printf("Lista Vazia!\n");

return NULL;

}

else{

do{

if(strcmp(n,aux->nome)<0)

return aux;

aux=aux->prox;

}while(aux!=L);

return aux->ant;

}

}

TContato \*busca\_contato\_nome\_posi(TIndice \*Li,char n[]){

TIndice \*posI;

TContato \*posC;

int i;

char nomeAux[100],nome2Aux[100],letra=n[0];

posI=verificar\_indice\_posi(Li,n);

if(posI!=NULL && posI->inicio!=NULL){// so entra se existir indice e se existir alguem cadastrado no indice

posC=posI->inicio;

for(i=0;n[i]!=0;i++){

nomeAux[i]=tolower(n[i]);

}

nomeAux[i]=0;

do{

for(i=0;i!=strlen(posC->nome);i++){

nome2Aux[i]=tolower(posC->nome[i]);

}

nome2Aux[i]=0;

if(strcmp(nomeAux,nome2Aux)==0)

return posC;

posC=posC->prox;

}while(posI->inicio!=posC);

}

return NULL;

}

TContato \*verificar\_contato(TContato \*L,char n[]){

}

void exibir\_fone(TIndice \*L){

TContato \*BuscaC;

char nome\_aux[100],op;

if(L==NULL){

printf("Lista vazia!\n");

}

else{

printf("\nDigite o nome :");

do{

gets(nome\_aux);fflush(stdin);

BuscaC = busca\_contato\_nome\_posi(L,nome\_aux);

if(BuscaC==NULL){

printf("Nome nao existente !\n ");

do{

printf("Deseja procurar outro nome? s - Sim\nn - Nao\n"

"Digite sua escolha :");

scanf("%c",&op);

}while(op != 's' && op != 'n');

if(op == 'n'){

printf("Procura cancelada !\n");

return;

}

}

}while(BuscaC==NULL);

printf("Contato %s \nTefone : %s",BuscaC->nome,BuscaC->fone);

}

}

void alterar\_fone(TIndice \*L){

TContato \*BuscaC;

char nome\_aux[100],op;

if(L==NULL){

printf("Lista vazia !\n");

}

else{

printf("\nDigite o nome :");

do{

gets(nome\_aux);fflush(stdin);

BuscaC = busca\_contato\_nome\_posi(L,nome\_aux);

if(BuscaC==NULL){

printf("Nome nao existente !\n ");

do{

printf("Deseja procurar outro nome? s - Sim\nn - Nao\n"

"Digite sua escolha :");

scanf("%c",&op);

}while(op != 's' && op != 'n');

if(op == 'n'){

printf("Procura cancelada !\n");

return;

}

}

}while(BuscaC==NULL);

printf("Digite o novo telefone: ");

gets(BuscaC->fone);

printf("Alteracao realizada com sucesso!\n");

}

}

void exibir\_agenda(TIndice \*L){

TContato \*aux\_c;

TIndice \*aux\_i=L;

if(L == NULL)

printf("Listas vazia!\n");

else{

printf("\nExibindo Lista completa : \n");

do{

printf("\n Letra %c:\n",aux\_i->letra);

aux\_c = aux\_i->inicio;

do{

printf(" nome : %s telefone : %s\n",aux\_c->nome,aux\_c->fone);

aux\_c = aux\_c ->prox;

}while(aux\_c!=aux\_i->inicio);

aux\_i = aux\_i ->prox;

}while(aux\_i!=L);

}

}

void remover\_contato(TIndice \*\*L){

TContato \*BuscaC,\*auxC;

TIndice \*pos\_i,\*auxI,\*aux\_LI=\*L;

char nome\_aux[100],op;

if(L==NULL){

printf("Lista Vazia!");

}

else{

printf("\nDigite o nome :");

do{

gets(nome\_aux);fflush(stdin);

BuscaC = busca\_contato\_nome\_posi(aux\_LI,nome\_aux);

if(BuscaC==NULL){

printf("Nome nao existente !\n ");

do{

printf("Deseja procurar outro nome? s - Sim\nn - Nao\n"

"Digite sua escolha :");

scanf("%c",&op);fflush(stdin);

}while(op != 's' && op != 'n');

if(op == 'n'){

printf("Procura cancelada !\nOperacao de remocao cancelada!\n");

return;

}

}

}while(BuscaC==NULL);

pos\_i=verificar\_indice(\*L,nome\_aux);

if((BuscaC->prox == pos\_i->inicio ) && BuscaC->ant == pos\_i->inicio){ // Quando houver somente 1 contato na lista do indice , remove os dois (indice e contato)

if(pos\_i->prox == pos\_i->ant){//se so existir um unico indice

pos\_i->inicio == NULL;

free(BuscaC);

free(pos\_i);

\*L = NULL;

}

else{// mais de um indices na lista

pos\_i->inicio == NULL;

free(BuscaC);

auxI = pos\_i->ant;

pos\_i->prox->ant=auxI;

auxI->prox=pos\_i->prox;

free(pos\_i);

}

}

else if(pos\_i->inicio == BuscaC){

auxC = BuscaC->ant;

auxC->prox = BuscaC->prox;

BuscaC->prox->ant = auxC;

pos\_i->inicio = BuscaC->prox;

free(BuscaC);

}

else{

auxC = BuscaC->ant;

auxC->prox = BuscaC->prox;

BuscaC->prox->ant = auxC;

free(BuscaC);

}

printf("Remocao realizada com sucesso!\n");

}

}

void limpar\_lista(TIndice \*\*L){

TIndice \*posUlt\_i,\*aux\_I;

TContato \*posUlt\_c,\*aux\_C;

if(\*L==NULL){

printf("Lista vazia!");

}

else{

posUlt\_i = (\*L)->ant;

posUlt\_c = posUlt\_i->inicio->ant;

if((posUlt\_i->ant == \*L && posUlt\_i->prox == \*L) && posUlt\_i->inicio->prox == posUlt\_i->inicio->ant){// caso com um so indice e um contato

aux\_C = posUlt\_i->inicio;

posUlt\_i->inicio=NULL;

free(aux\_C);

\*L = NULL;

free(posUlt\_i);

}

else{

do{

do{

aux\_C= posUlt\_c->prox;

posUlt\_c->ant->prox = aux\_C;

aux\_C->ant=posUlt\_c->ant;

free(posUlt\_c);

posUlt\_c=aux\_C->ant;

}while(posUlt\_c!=posUlt\_i->inicio);

if((posUlt\_i->ant == \*L && posUlt\_i->prox == \*L) && posUlt\_c==aux\_C)

break;

posUlt\_i->inicio==NULL;

free(posUlt\_c);

aux\_I = posUlt\_i->prox;

posUlt\_i -> ant ->prox= aux\_I;

aux\_I->ant=posUlt\_i->ant;

free(posUlt\_i);

posUlt\_i = posUlt\_i->ant;

}while(posUlt\_i!=\*L);

aux\_C = posUlt\_i->inicio;

posUlt\_i->inicio=NULL;

free(aux\_C);

\*L = NULL;

free(posUlt\_i);

}

printf("Lista limpa com sucesso!\n");

}

}