Universidade Federal Fluminense

Matéria: Estrutura de Dados

Professor: Dalessandro Soares

Aluno: Alessandro Sampaio

#include <stdio.h>

#include <stdlib.h>

typedef struct lista{

int info;

struct lista \*prox;

}Lista;

Lista\* inserir\_inicio (Lista \*l, int y){

Lista \*l1=l, \*p=l;

Lista \*novo=(Lista\*)malloc(sizeof(Lista));

novo->info=y;

l1=novo;

l1->prox=p;

return l1;

}

Lista\* inserir\_fim(Lista \*l, int y){

if (l == NULL){

return l=inserir\_inicio(l,y);

}

else{

l->prox=inserir\_fim(l->prox,y);

return l;

}

}

Lista\* remover\_lista(Lista \*l, int y){

Lista \*l1;

if (l->info == y){

l1=l;

l=l->prox;

free(l1);

}

else

l->prox=remover\_lista (l->prox,y);

return l;

}

Lista\* altera\_lista(Lista \*l, int y){

int x;

if (l->info == y){

printf ("Digite o novo numero\n");

scanf ("%d",&x);

l->info=x;

}

else

l->prox=altera\_lista(l->prox,y);

return l;

}

void imprimir\_lista (Lista\*l){

if (l==NULL)

{

}

else{

printf ("%d ",l->info);

imprimir\_lista(l->prox);

}

}

int verifica\_lista(Lista \*l, int y){

if(l==NULL)

return 0;

else{

if(l->info==y)

return 1;

verifica\_lista(l->prox,y);

}

}

int conta\_par\_lista(Lista \*l){

if(l==NULL)

return 0;

else{

if(l->info%2==0)

return (1+conta\_par\_lista(l->prox));

else

return (0+conta\_par\_lista(l->prox));

}

}

Lista\* remove\_lista(Lista \*l){

if(l->prox==NULL)

free(l);

else

l->prox=remove\_lista(l->prox);

}

int main(void)

{

int x=1,y=0,z=0,w=0;

Lista \*l=NULL;

while (x != 9)

{

system("cls");

printf ("Digite a opcao desejada\n\n");

printf ("1 - Inserir um elemento no incio da lista\n");

printf ("2 - Inserir um elemento no final da lista\n");

printf ("3 - Remover um elemento da lista\n");

printf ("4 - Alterar um elemento da lista\n");

printf ("5 - Imprimir a lista\n");

printf ("6 - Verificar se existe o elemento na lista\n");

printf ("7 - Contar elementos pares na lista\n");

printf ("8 - Sair\n\n");

scanf ("%d",&x);

system("cls");

switch (x)

{

case 1:

{

printf ("Digite o numero a ser inserido no inicio da lista\n");

scanf ("%d",&y);

l=inserir\_inicio(l,y);

break;

}

case 2:

{

printf ("Digite o numero a ser inserido no fim da lista\n");

scanf ("%d",&y);

l=inserir\_fim(l,y);

break;

}

case 3:

{

printf ("Digite o numero a ser removido da lista\n");

scanf ("%d",&y);

z=verifica\_lista(l,y);

if(z==1)

{

l=remover\_lista(l,y);

printf ("O numero %d foi removido\n",y);

system("pause");

}

else{

printf ("O numero %d nao existe na lista\n",y);

system("pause");

}

break;

}

case 4:

{

printf ("Digite o numero a ser alterado\n");

scanf ("%d",&y);

z=verifica\_lista(l,y);

if(z==1){

l=altera\_lista(l,y);

printf ("O numero foi alterado\n");

system("pause");

}

else{

printf ("O numero %d nao existe na lista\n",y);

system("pause");

}

break;

}

case 5:

{

printf ("Sua lista: ");

imprimir\_lista(l);

printf ("\n");

system("pause");

break;

}

case 6:

{

printf ("Digite o elemento a ser verificado\n");

scanf ("%d",&y);

z=verifica\_lista(l,y);

printf(" Retorno: %d\n",z);

if(z==0)

printf("Numero nao encontrado\n");

else

printf("Numero encontrado\n");

system("pause");

break;

}

case 7:

{

w=conta\_par\_lista(l);

printf("A quantidade de numeros par: %d\n",w);

system("pause");

break;

}

case 8:

{

if(l!=NULL)

l=remove\_lista(l);

x=9;

}

}

}

return 0;

}