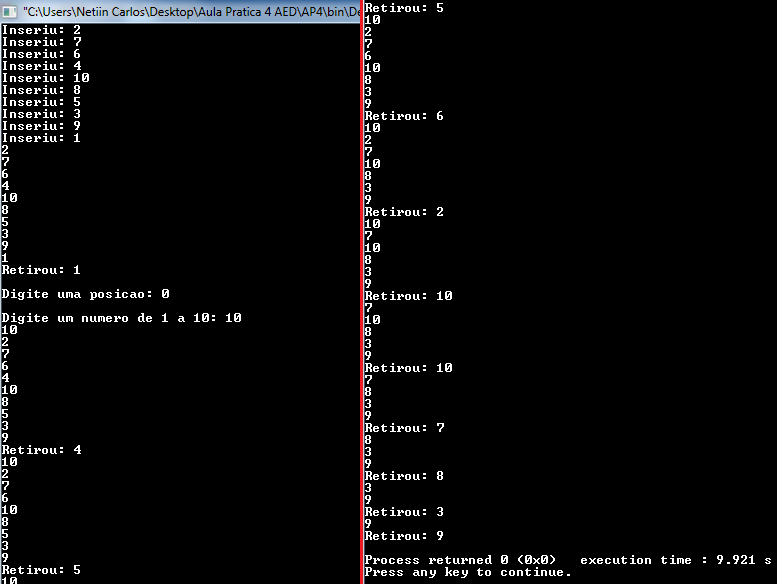
UFU

Nome: Antonio Carlos Neto

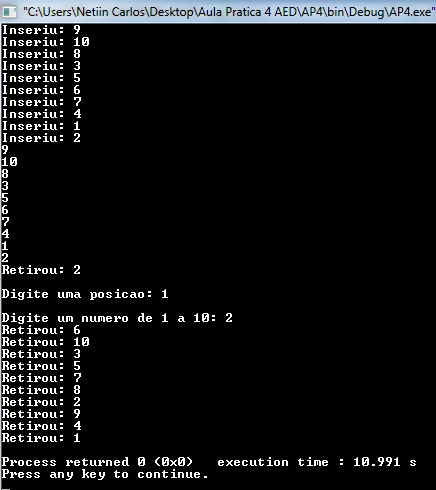
Curso: Ciência da Computação

Disciplina: GBC024

EXECUÇÃO COM TESTE:



EXECUÇÃO SEM TESTE:



Main.c :

#include <sys/time.h>

#include <stdio.h>

#include <stdlib.h>

#include "listaArray.h"

int main(int argc, char \*argv[])

{

struct timeval t;

TipoLista lista;

TipoItem item, x2;

int vetor[MAXTAM];

int i, j, k, n, max,x1;

gettimeofday(&t,NULL);

srand((unsigned int)t.tv\_usec);

max = 10;

FLVazia(&lista);

for(i = 0; i < max; i++) vetor[i] = i + 1;

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

{

k = (int) (10.0 \* rand()/(RAND\_MAX + 1.0));

j = (int) (10.0 \* rand()/(RAND\_MAX + 1.0));

n = vetor[k];

vetor[k] = vetor[j];

vetor[j] = n;

}

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

{

item.Chave = vetor[i];

Insere(item, &lista);

printf("Inseriu: %d \n", item.Chave);

}

Imprime(lista);

Retira(9, &lista, &item);

printf("Retirou: %d\n", item.Chave);

printf("\nDigite uma posicao: ");

do

{

scanf("%d",&x1);

}

while(x1>9 || x1<0);

printf("\nDigite um numero de 1 a 10: ");

do

{

scanf("%d",&x2.Chave);

}

while(x2.Chave >10 || x2.Chave <= 0);

item.Chave = x2.Chave;

Inserex(x1,item, &lista);

///////////////TESTE//////////////Imprime (lista);

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

{

j = (int) ((lista.Ultimo - 1) \* rand() / (RAND\_MAX + 1.0));

Retira(j, &lista, &item);

printf("Retirou: %d\n", item.Chave);

////////////////TESTE/////////////////Imprime(lista);

}

Imprime (lista);

return(0);

}

listaArray.c :

#include <stdio.h>

#include <stdlib.h>

#include "listaArray.h"

void FLVazia(TipoLista \*Lista)

{

Lista -> Ultimo = -1;

}

int Vazia(TipoLista Lista)

{

return (Lista.Ultimo == -1);

}

void Insere(TipoItem x, TipoLista \*Lista)

{

if (Lista -> Ultimo == MAXTAM - 1) printf("Lista esta cheia\n");

else

{

Lista -> Ultimo++;

Lista -> Item[Lista -> Ultimo] = x;

}

} /\* Insere \*/

void Inserex(int n,TipoItem x, TipoLista \*Lista)

{

int i;

if (Lista -> Ultimo == MAXTAM - 1) printf("Lista esta cheia\n");

else

{

Lista -> Ultimo++;

for(i=Lista -> Ultimo; i>n ; i--)

{

Lista -> Item[i] = Lista -> Item[i -1];

}

Lista -> Item[n] = x;

}

}

void Retira(TipoApontador p, TipoLista \*Lista, TipoItem \*Item)

{

int Aux;

if (Vazia(\*Lista) || p > Lista -> Ultimo)

{

printf(" Erro Posicao nao existe\n");

return;

}

\*Item = Lista -> Item[p];

for (Aux = p + 1; Aux <= Lista -> Ultimo; Aux++)

Lista -> Item[Aux - 1] = Lista -> Item[Aux];

Lista -> Ultimo--;

} /\* Retira \*/

void Imprime(TipoLista Lista)

{

int Aux;

for (Aux = 0; Aux <= (Lista.Ultimo); Aux++)

printf("%d\n", Lista.Item[Aux].Chave);

} /\* Imprime \*/

listaArray.h :

#ifndef LISTAARRAY\_H\_INCLUDED

#define LISTAARRAY\_H\_INCLUDED

#define MAXTAM 1000

typedef int TipoChave;

typedef int TipoApontador;

typedef struct

{

TipoChave Chave;

} TipoItem;

typedef struct

{

TipoItem Item[MAXTAM];

TipoApontador Ultimo;

} TipoLista;

void FLVazia(TipoLista \*Lista);

int Vazia(TipoLista Lista);

void Insere(TipoItem x, TipoLista \*Lista);

void Retira(TipoApontador p, TipoLista \*Lista, TipoItem \*Item);

void Imprime(TipoLista Lista);

void Inserex(int n,TipoItem x, TipoLista \*Lista);

#endif // LISTAARRAY\_H\_INCLUDED