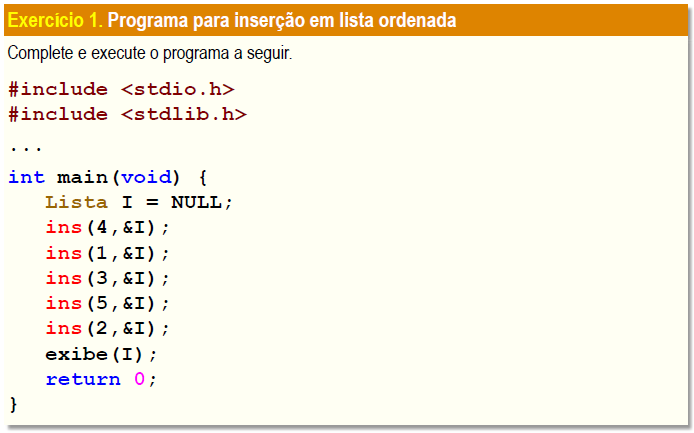
ed-09 – 21/10/2021



#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct no {

Item item;

struct no \*prox;

} \*Lista;

Lista no(Item x, Lista p) {

Lista n = malloc(sizeof(struct no));

n -> item = x;

n -> prox = p;

return;

}

void ins(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

\*L = no(x,\*L);

}

void exibe(Lista L) {

printf("[");

while(L != NULL) {

printf("%d, ", L -> item);

L = L -> prox;

}

printf("\b\b]");

}

int main(void) {

Lista I = NULL;

ins(4, &I);

ins(1, &I);

ins(3, &I);

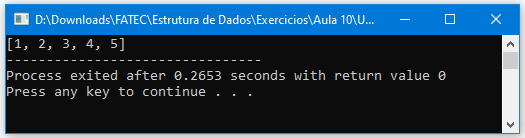
ins(5, &I);

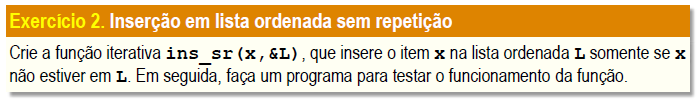
ins(2, &I);

exibe(I);

return 0;

}





#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

typedef int Item;

typedef struct no {

Item item;

struct no \*prox;

} \*Lista;

Lista no(Item x, Lista p) {

Lista n = malloc(sizeof(struct no));

n -> item = x;

n -> prox = p;

return;

}

void ins(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

\*L = no(x,\*L);

}

void exibe(Lista L) {

printf("[");

while(L != NULL) {

printf("%d, ", L -> item);

L = L -> prox;

}

printf("\b\b]");

}

bool pertence(int r, Lista L) {

int i = 0;

while(L != NULL){

i = L -> item;

if(r == i){

return 1;

}

L = L -> prox;

}

return 0;

}

void ins\_sr(int x, Lista L) {

if(pertence(x, L) == 1) puts("Numero ja na lista.");

else ins(x, &L);

}

int main(void) {

int x, ask=0;

Lista I = NULL;

ins(4, &I);

ins(1, &I);

ins(3, &I);

ins(5, &I);

ins(2, &I);

exibe(I);

do {

printf("\n\nNumero a ser inserido:... ");

scanf("%d", &x);

ins\_sr(x, I);

printf("Deseja continuar inserindo?...\n[1] - sim\n[2] - nao\n");

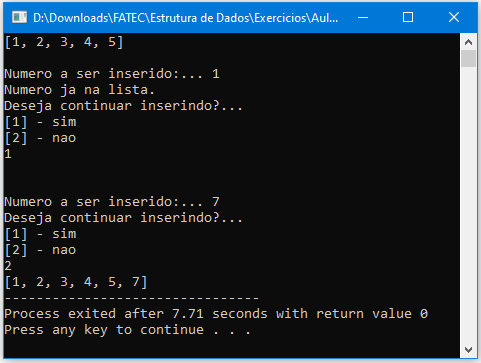
scanf("%d", &ask);

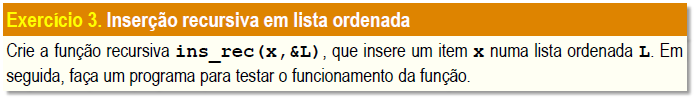
}while(ask == 1);

exibe(I);

return 0;

}





#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

typedef int Item;

typedef struct no {

Item item;

struct no \*prox;

} \*Lista;

Lista no(Item x, Lista p) {

Lista n = malloc(sizeof(struct no));

n -> item = x;

n -> prox = p;

return;

}

void ins(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

\*L = no(x,\*L);

}

void exibe(Lista L) {

printf("[");

while(L != NULL) {

printf("%d, ", L -> item);

L = L -> prox;

}

printf("\b\b]");

}

bool pertence(int r, Lista L) {

int i = 0;

while(L != NULL){

i = L -> item;

if(r == i){

return 1;

}

L = L -> prox;

}

return 0;

}

void ins\_rec(Item x, Lista \*L) {

if (\*L != NULL && (\*L)->item < x){

ins\_rec(x, &(\*L)->prox);

}else{

\*L = no(x,\*L);

}

}

int main(void) {

int x, ask=0;

Lista I = NULL;

ins(4, &I);

ins(1, &I);

ins(3, &I);

ins(5, &I);

ins(2, &I);

exibe(I);

do {

printf("\n\nNumero a ser inserido:... ");

scanf("%d", &x);

ins\_rec(x, &I);

printf("Deseja continuar inserindo?...\n[1] - sim\n[2] - nao\n");

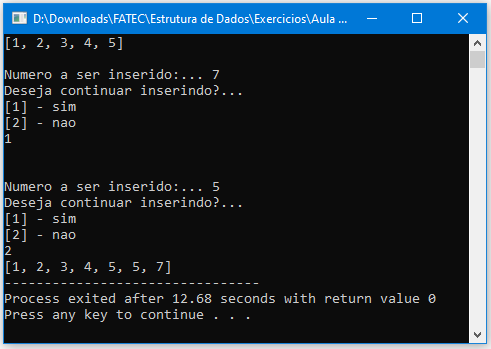
scanf("%d", &ask);

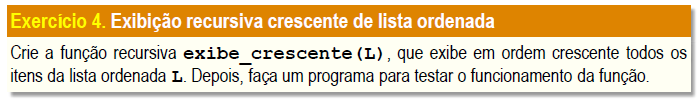
}while(ask == 1);

exibe(I);

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct no {

Item item;

struct no \*prox;

} \*Lista;

Lista no(Item x, Lista p) {

Lista n = malloc(sizeof(struct no));

n -> item = x;

n -> prox = p;

return;

}

void ins(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

\*L = no(x,\*L);

}

void exibe(Lista L) {

printf("[");

while(L != NULL) {

printf("%d, ", L -> item);

L = L -> prox;

}

printf("\b\b]");

}

void exibe\_crescente (Lista L) {

if(L == NULL) return 0;

printf("%d, ", L -> item);

L = L -> prox;

exibe\_crescente(L);

}

int main(void) {

Lista I = NULL;

ins(4, &I);

ins(1, &I);

ins(3, &I);

ins(5, &I);

ins(2, &I);

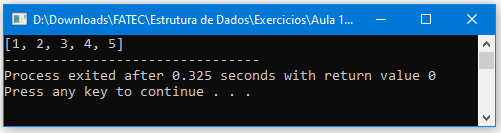
printf("[");

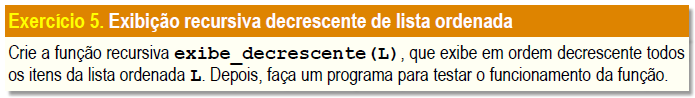
exibe\_crescente(I);

printf("\b\b]");

return 0;

}





#include <stdio.h>

#include <stdlib.h>

#include "pilhaInt.h"

typedef int Item;

typedef struct no {

Item item;

struct no \*prox;

} \*Lista;

Lista no(Item x, Lista p) {

Lista n = malloc(sizeof(struct no));

n -> item = x;

n -> prox = p;

return;

}

void ins(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

\*L = no(x,\*L);

}

void exibe(Lista L) {

printf("[");

while(L != NULL) {

printf("%d, ", L -> item);

L = L -> prox;

}

printf("\b\b]");

}

void exibe\_decrescente (Lista L) {

Pilha P = pilha(7);

int n = 0;

if(L == NULL) return 0;

n = L -> item;

empilha(n, P);

L = L -> prox;

exibe\_decrescente(L);

printf("%d, ", desempilha(P));

}

int main(void) {

Lista I = NULL;

ins(4, &I);

ins(1, &I);

ins(3, &I);

ins(5, &I);

ins(2, &I);

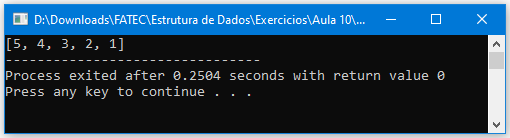
printf("[");

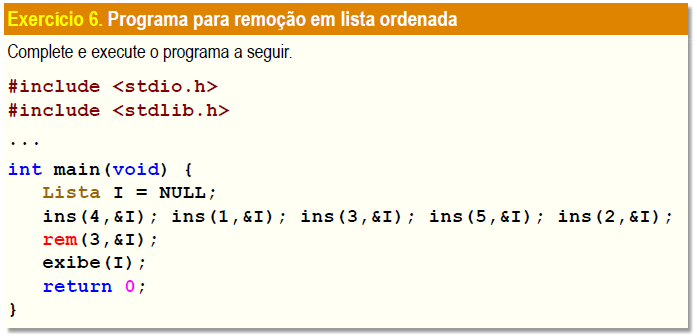
exibe\_decrescente(I);

printf("\b\b]");

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct no {

Item item;

struct no \*prox;

} \*Lista;

Lista no(Item x, Lista p) {

Lista n = malloc(sizeof(struct no));

n -> item = x;

n -> prox = p;

return;

}

void ins(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

\*L = no(x,\*L);

}

void exibe(Lista L) {

printf("[");

while(L != NULL) {

printf("%d, ", L -> item);

L = L -> prox;

}

printf("\b\b]");

}

void rem(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

if(\*L == NULL || (\*L) -> item > x) return;

Lista n = \*L;

\*L = n -> prox;

free(n);

}

int main(void) {

Lista I = NULL;

ins(4, &I);

ins(1, &I);

ins(3, &I);

ins(5, &I);

ins(2, &I);

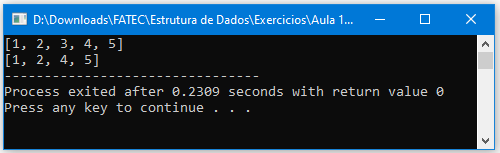
exibe(I); printf("\n");

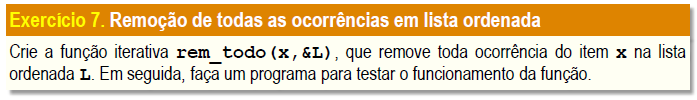
rem(3, &I);

exibe(I);

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct no {

Item item;

struct no \*prox;

} \*Lista;

Lista no(Item x, Lista p) {

Lista n = malloc(sizeof(struct no));

n -> item = x;

n -> prox = p;

return;

}

void ins(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

\*L = no(x,\*L);

}

void exibe(Lista L) {

printf("[");

while(L != NULL) {

printf("%d, ", L -> item);

L = L -> prox;

}

printf("\b\b]");

}

void rem(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

if(\*L == NULL || (\*L) -> item > x) return;

Lista n = \*L;

\*L = n -> prox;

free(n);

}

void rem\_todo(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

if(\*L == NULL || (\*L) -> item > x) return;

while(\*L != NULL && (\*L) -> item == x) {

Lista n = \*L;

\*L = n -> prox;

free(n);

}

}

int main(void) {

Lista I = NULL;

ins(4, &I);

ins(1, &I);

ins(1, &I);

ins(3, &I);

ins(5, &I);

ins(1, &I);

ins(2, &I);

ins(5, &I);

exibe(I); printf("\n");

rem(3, &I);

exibe(I); printf("\n");

rem\_todo(1, &I);

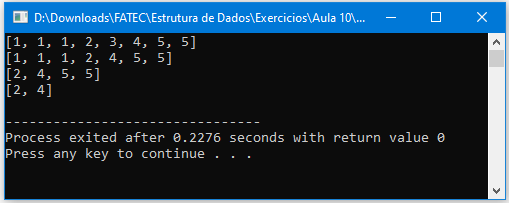
exibe(I); printf("\n");

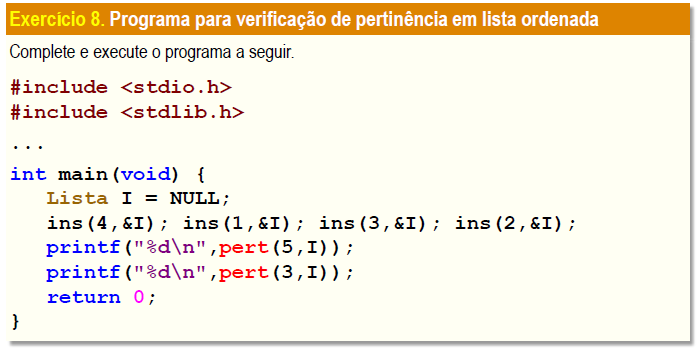
rem\_todo(5, &I);

exibe(I); printf("\n");

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct no {

Item item;

struct no \*prox;

} \*Lista;

Lista no(Item x, Lista p) {

Lista n = malloc(sizeof(struct no));

n -> item = x;

n -> prox = p;

return;

}

void ins(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

\*L = no(x,\*L);

}

int pert(Item x, Lista L) {

while(L != NULL && L -> item < x)

L = L -> prox;

return(L != NULL && L -> item == x);

}

int main(void) {

Lista I = NULL;

ins(4, &I);

ins(1, &I);

ins(3, &I);

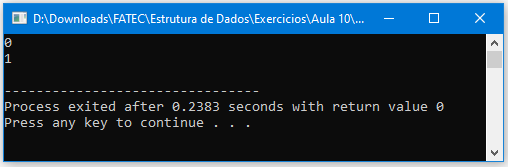
ins(2, &I);

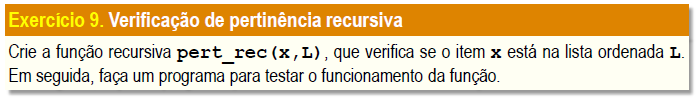
printf("%d\n",pert(5,I));

printf("%d\n",pert(3,I));

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct no {

Item item;

struct no \*prox;

} \*Lista;

Lista no(Item x, Lista p) {

Lista n = malloc(sizeof(struct no));

n -> item = x;

n -> prox = p;

return;

}

void ins(Item x, Lista \*L) {

while(\*L != NULL && (\*L) -> item < x)

L = &(\*L) -> prox;

\*L = no(x,\*L);

}

int pert\_rec(Item x, Lista L) {

if(L != NULL && L -> item < x){

L = L -> prox;

return pert\_rec(x, L);

}

return(L != NULL && L -> item == x);

}

int main(void) {

Lista I = NULL;

ins(4, &I);

ins(1, &I);

ins(3, &I);

ins(2, &I);

printf("%d\n",pert\_rec(4,I));

printf("%d\n",pert\_rec(5,I));

printf("%d\n",pert\_rec(3,I));

return 0;

}

