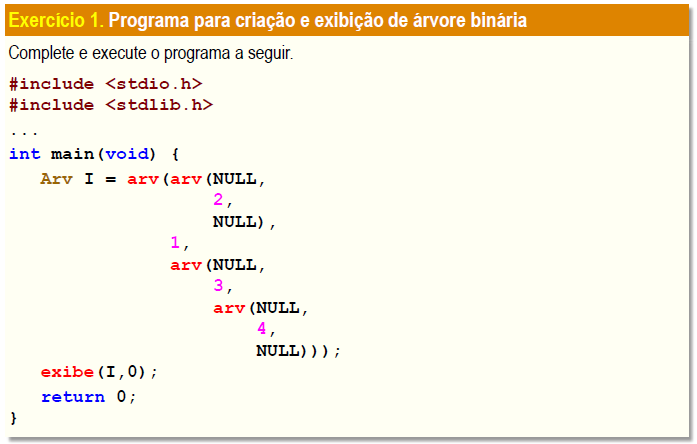
ed-010 – 21/10/2021



#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv(Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n -> esq = e;

n -> item = x;

n -> dir = d;

return n;

}

void exibe(Arv A, int n) {

if(A == NULL) return;

exibe(A -> dir, n+1);

printf("%\*s%d\n", 3\*n, "", A -> item);

exibe(A -> esq, n+1);

}

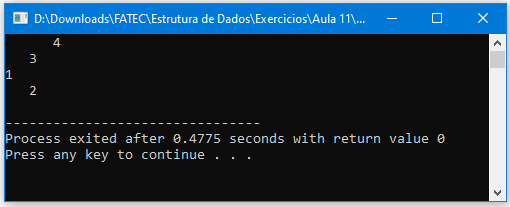
int main(void) {

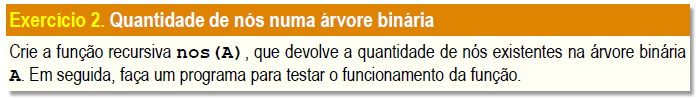
Arv I = arv(arv(NULL, 2, NULL), 1, arv(NULL, 3, arv(NULL, 4, NULL)));

exibe(I, 0);

return 0;

}





#include <stdio.h>

#include <stdlib.h>

int n = 0;

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv(Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n -> esq = e;

n -> item = x;

n -> dir = d;

return n;

}

void exibe(Arv A, int n) {

if(A == NULL) return;

exibe(A -> dir, n+1);

printf("%\*s%d\n", 3\*n, "", A -> item);

exibe(A -> esq, n+1);

}

int nos(Arv A) {

if(A == NULL) return;

if(A -> dir != NULL) {

n += 1;

nos(A -> dir);

}

if(A -> esq != NULL) {

n += 1;

nos(A -> esq);

}

return n + 1;

}

int main(void) {

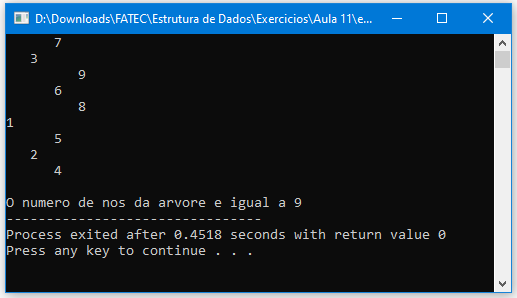
Arv I = arv(arv(arv(NULL, 4, NULL), 2, arv(NULL, 5, NULL)), 1, arv(arv(arv(NULL, 8, NULL), 6, arv(NULL, 9, NULL)), 3, arv(NULL, 7, NULL)));

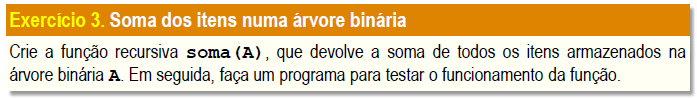
exibe(I, 0);

printf("\nO numero de nos da arvore e igual a %d", nos(I));

return 0;

}





#include <stdio.h>

#include <stdlib.h>

int n = 0, s = 0;

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv(Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n -> esq = e;

n -> item = x;

n -> dir = d;

return n;

}

void exibe(Arv A, int n) {

if(A == NULL) return;

exibe(A -> dir, n+1);

printf("%\*s%d\n", 3\*n, "", A -> item);

exibe(A -> esq, n+1);

}

int soma(Arv A) {

if(A == NULL) return;

n = A -> item;

s += n;

if(A -> dir != NULL) {

soma(A -> dir);

}

if(A -> esq != NULL) {

soma(A -> esq);

}

return s;

}

int main(void) {

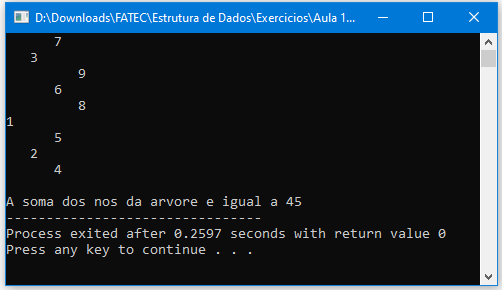
Arv I = arv(arv(arv(NULL, 4, NULL), 2, arv(NULL, 5, NULL)), 1, arv(arv(arv(NULL, 8, NULL), 6, arv(NULL, 9, NULL)), 3, arv(NULL, 7, NULL)));

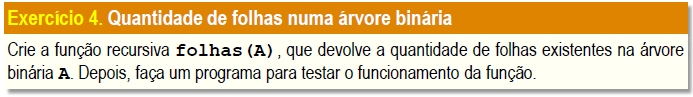
exibe(I, 0);

printf("\nA soma dos nos da arvore e igual a %d", soma(I));

return 0;

}





#include <stdio.h>

#include <stdlib.h>

int f = 0;

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv(Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n -> esq = e;

n -> item = x;

n -> dir = d;

return n;

}

void exibe(Arv A, int n) {

if(A == NULL) return;

exibe(A -> dir, n+1);

printf("%\*s%d\n", 3\*n, "", A -> item);

exibe(A -> esq, n+1);

}

int folhas(Arv A) {

if(A == NULL) return;

if(A -> dir != NULL) {

folhas(A -> dir);

}

if(A -> esq != NULL) {

folhas(A -> esq);

}

if(A -> dir == NULL && A -> esq == NULL) f += 1;

return f;

}

int main(void) {

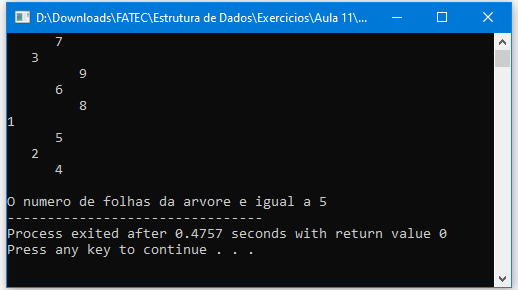
Arv I = arv(arv(arv(NULL, 4, NULL), 2, arv(NULL, 5, NULL)), 1, arv(arv(arv(NULL, 8, NULL), 6, arv(NULL, 9, NULL)), 3, arv(NULL, 7, NULL)));

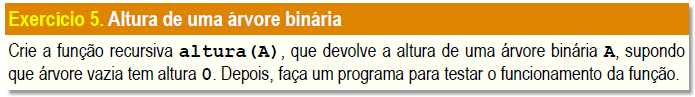
exibe(I, 0);

printf("\nO numero de folhas da arvore e igual a %d", folhas(I));

return 0;

}





#include <stdio.h>

#include <stdlib.h>

int dir = 0, esq = 0;

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv(Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n -> esq = e;

n -> item = x;

n -> dir = d;

return n;

}

void exibe(Arv A, int n) {

if(A == NULL) return;

exibe(A -> dir, n+1);

printf("%\*s%d\n", 3\*n, "", A -> item);

exibe(A -> esq, n+1);

}

int altura(Arv A) {

if(A == NULL) return;

if(A -> dir != NULL) {

dir += 1;

altura(A -> dir);

}

if(A -> esq != NULL) {

esq += 1;

altura(A -> esq);

}

if(dir > esq) return dir;

else return esq;

}

int main(void) {

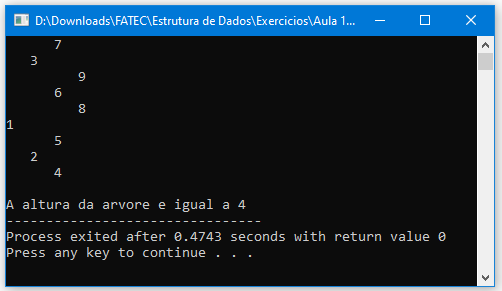
Arv I = arv(arv(arv(NULL, 4, NULL), 2, arv(NULL, 5, NULL)), 1, arv(arv(arv(NULL, 8, NULL), 6, arv(NULL, 9, NULL)), 3, arv(NULL, 7, NULL)));

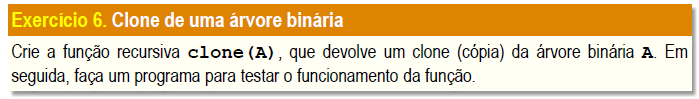
exibe(I, 0);

printf("\nA altura da arvore e igual a %d", altura(I));

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv (Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n->esq = e;

n->item = x;

n->dir = d;

return n;

}

Arv clone (Arv A) {

if ( A == NULL ) return NULL;

return arv(clone(A->esq),A->item,clone(A->dir));

}

void exibe (Arv A, int n) {

if (A==NULL) return;

exibe(A->dir, n+1);

printf("%\*s%d\n", 3\*n, "", A->item);

exibe(A->esq, n+1);

}

int main(void) {

Arv I = arv(arv(arv(NULL, 4, NULL), 2, arv(NULL, 5, NULL)), 1, arv(arv(arv(NULL, 8, NULL), 6, arv(NULL, 9, NULL)), 3, arv(NULL, 7, NULL)));

printf("Arv I\n");

exibe(I, 0);

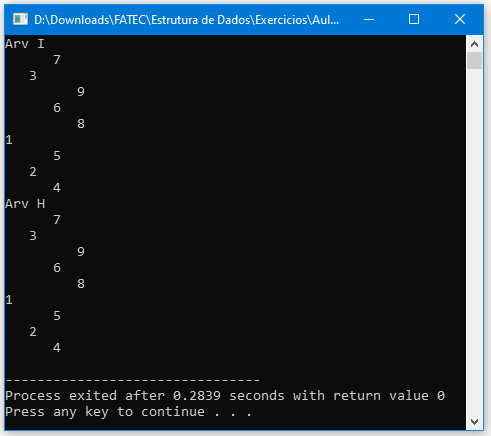
printf("Arv H\n");

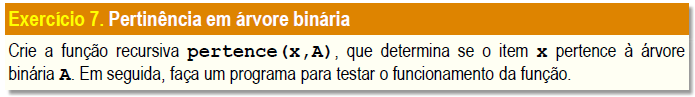
Arv H = clone (I);

exibe(H, 0);

return 0;

}





#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv(Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n -> esq = e;

n -> item = x;

n -> dir = d;

return n;

}

void exibe(Arv A, int n) {

if(A == NULL) return;

exibe(A -> dir, n+1);

printf("%\*s%d\n", 3\*n, "", A -> item);

exibe(A -> esq, n+1);

}

bool pertence(int x, Arv A) {

if(A == NULL) return false;

if(x == A -> item) return true;

if(A -> dir != NULL) {

return pertence(x, A -> dir);

}

if(A -> esq != NULL) {

return pertence(x, A -> esq);

}

}

int main(void) {

int n = 0;

Arv I = arv(arv(arv(NULL, 4, NULL), 2, arv(NULL, 5, NULL)), 1, arv(arv(arv(NULL, 8, NULL), 6, arv(NULL, 9, NULL)), 3, arv(NULL, 7, NULL)));

exibe(I, 0);

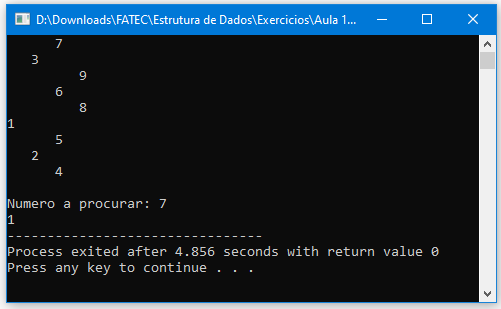
printf("\nNumero a procurar: ");

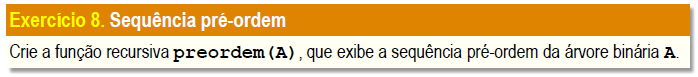
scanf("%d", &n);

printf("%d", pertence(n, I));

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv (Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n->esq = e;

n->item = x;

n->dir = d;

return n;

}

void preOrdem (Arv A){

if ( A != NULL ){

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

preOrdem(A->esq);

preOrdem(A->dir);

}

}

void exibe (Arv A, int n) {

if (A==NULL) return;

exibe(A->dir, n+1);

printf("%\*s%d\n", 3\*n, "", A->item);

exibe(A->esq, n+1);

}

int main (void){

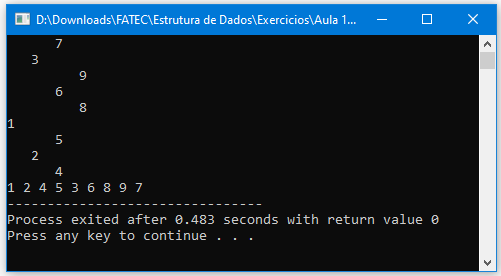
Arv I = arv(arv(arv(NULL, 4, NULL), 2, arv(NULL, 5, NULL)), 1, arv(arv(arv(NULL, 8, NULL), 6, arv(NULL, 9, NULL)), 3, arv(NULL, 7, NULL)));

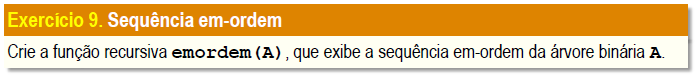
exibe(I, 0);

preOrdem(I);

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv (Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n->esq = e;

n->item = x;

n->dir = d;

return n;

}

void emOrdem (Arv A){

if ( A != NULL ){

emOrdem(A->esq);

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

emOrdem(A->dir);

}

}

void exibe (Arv A, int n) {

if (A==NULL) return;

exibe(A->dir, n+1);

printf("%\*s%d\n", 3\*n, "", A->item);

exibe(A->esq, n+1);

}

int main (void){

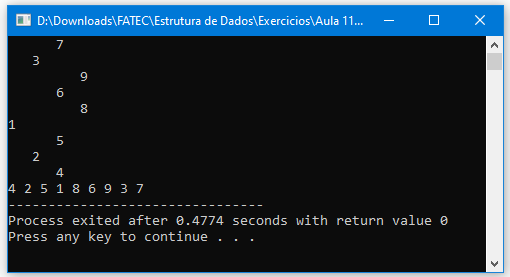
Arv I = arv(arv(arv(NULL, 4, NULL), 2, arv(NULL, 5, NULL)), 1, arv(arv(arv(NULL, 8, NULL), 6, arv(NULL, 9, NULL)), 3, arv(NULL, 7, NULL)));

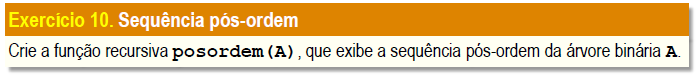
exibe(I, 0);

emOrdem(I);

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv (Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n->esq = e;

n->item = x;

n->dir = d;

return n;

}

void posOrdem (Arv A){

if ( A != NULL ){

posOrdem(A->esq);

posOrdem(A->dir);

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

}

}

void exibe (Arv A, int n) {

if (A==NULL) return;

exibe(A->dir, n+1);

printf("%\*s%d\n", 3\*n, "", A->item);

exibe(A->esq, n+1);

}

int main (void){

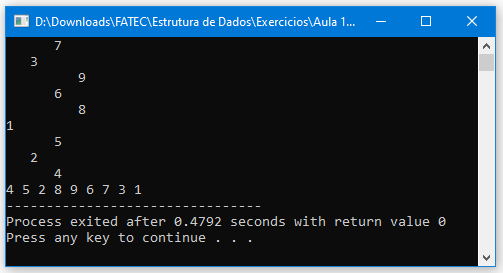
Arv I = arv(arv(arv(NULL, 4, NULL), 2, arv(NULL, 5, NULL)), 1, arv(arv(arv(NULL, 8, NULL), 6, arv(NULL, 9, NULL)), 3, arv(NULL, 7, NULL)));

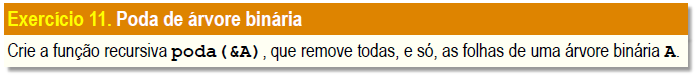
exibe(I, 0);

posOrdem(I);

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv (Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n->esq = e;

n->item = x;

n->dir = d;

return n;

}

Item poda(Arv \*A) {

if( \*A == NULL ) abort();

while( (\*A)->dir != NULL ) A = &(\*A)->dir;

Arv n = \*A;

Item x = n->item;

\*A = n->esq;

free(n);

return x;

}

void exibe (Arv A, int n) {

if (A==NULL) return;

exibe(A->dir, n+1);

printf("%\*s%d\n", 3\*n, "", A->item);

exibe(A->esq, n+1);

}

int main (void){

Arv I = arv(arv(NULL, 2, NULL), 1, arv(NULL, 3, arv(NULL, 4, NULL)));

exibe(I, 0);

printf("Podando as folhas da arvore...\n");

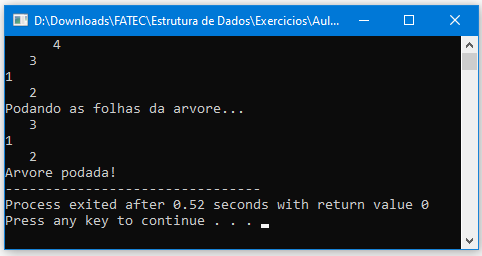
poda(&I);

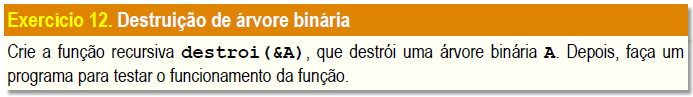
exibe(I, 0);

printf("Arvore podada!");

return 0;

}





#include <stdio.h>

#include <stdlib.h>

typedef int Item;

typedef struct arv {

struct arv \*esq;

Item item;

struct arv \*dir;

} \*Arv;

Arv arv (Arv e, Item x, Arv d) {

Arv n = malloc(sizeof(struct arv));

n->esq = e;

n->item = x;

n->dir = d;

return n;

}

void destroi(Arv \*A) {

if( \*A == NULL ) return;

destroi(&(\*A)->esq);

destroi(&(\*A)->dir);

free(\*A);

\*A = NULL;

}

void exibe (Arv A, int n) {

if (A==NULL) return;

exibe(A->dir, n+1);

printf("%\*s%d\n", 3\*n, "", A->item);

exibe(A->esq, n+1);

}

int main (void){

Arv I = arv(arv(NULL, 2, NULL), 1, arv(NULL, 3, arv(NULL, 4, NULL)));

exibe(I, 0);

printf("Destruindo arvore...\n");

destroi(&I);

exibe(I, 0);

printf("Arvore destruida!");

return 0;

}

