include <stdio.h>

#include <stdlib.h>

int quantidadeNos = 0;

typedef struct \_no{

int chave;

int cont;

struct \_no \*esq, \*dir, \*pai;

}no;

no\* auxPai = NULL;

void vaziaArvore(no \*raiz){

if (raiz == NULL){

printf(" A arvore esta vazia\n\n");

}

else {

printf(" A arvore nao esta vazia\n\n");

}

}

void insere (int x, no \*\*p){

if (\*p==NULL){

\*p=(no \*)malloc(sizeof(no));

(\*p)->chave=x;

(\*p)->dir=NULL;

(\*p)->esq=NULL;

(\*p)->pai = auxPai;

(\*p)->cont=0;

(\*p)->cont++;

}

else{

if (x<(\*p)->chave){

if((\*p)->esq == NULL)

auxPai = \*p;

insere(x, &(\*p)->esq);

}

if(x>(\*p)->chave){

if((\*p)->dir == NULL)

auxPai = \*p;

insere(x, &(\*p)->dir);

}

if(x == (\*p)->chave){

if((\*p)->dir == NULL)

auxPai = \*p;

insere(x, &(\*p)->dir);

}

}

}

int contarNos(no \*p){

if(p == NULL)

return 0;

else

return 1 + contarNos(p->esq) + contarNos(p->dir);

}

no\* busca(no \*arvore, int x){

if (arvore == NULL)

return NULL;

if (x == arvore->chave)

return arvore;

if (x < arvore->chave)

return busca(arvore->esq, x);

else

return busca(arvore->dir, x);

}

void consultarValor(no\* raiz){

int num;

no\* aux;

printf("Digite o numero procurado: ");

scanf("%d", &num);

aux = busca(raiz, num);

if (aux == NULL)

printf("Nao encontrado!\n");

else{

printf("Encontrado!\n");

printf("O numero %d foi repetido %d vezes\n", num, aux->cont);

}

}

void consultarDetalhes(no\* raiz){

int num;

no\* aux;

printf("Digite o numero procurado: ");

scanf("%d", &num);

aux = busca(raiz, num);

if (aux == NULL)

printf("Valor Nao encontrado!\n");

else{

printf("Valor Encontrado!\n");

if(aux->pai == NULL){

printf("O No que contem o valor %d e a raiz da arvore.\n", num);

if(aux->esq != NULL)

printf("Valor contido no No filho a esquerda: %d\n", aux->esq->chave);

if(aux->dir != NULL)

printf("Valor contido no No filho a direito: %d\n", aux->dir->chave);

}

if((aux->esq == NULL)&&(aux->dir == NULL)&&(aux->pai != NULL)){

printf("O No que contem o valor %d e uma folha da arvore.\n", num);

printf("Valor contido no No pai: %d\n", aux->pai->chave);

}

if(((aux->esq != NULL)||(aux->dir != NULL))&&(aux->pai != NULL)){

printf("O No que contem o valor %d e um no interno a arvore.\n", num);

printf("Valor contido no No pai: %d\n", aux->pai->chave);

if(aux->esq != NULL)

printf("Valor contido no No filho a esquerda: %d\n", aux->esq->chave);

if(aux->dir != NULL)

printf("Valor contido no No filho a direito: %d\n", aux->dir->chave);

}

}

}

void imprime(no \*p, int nivel){

int i;

if(p == NULL)

return;

imprime(p->dir, nivel+1);

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

printf(" ");

printf("%6d\n\n",p->chave);

imprime(p->esq,nivel+1);

}

void consultaNivel(no \*p){

int i, nivel;

if(p == NULL)

return;

imprime(p->dir, nivel+1);

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

printf(" ");

printf("%6d\n\n",p->chave);

imprime(p->esq,nivel+1);

}

void preordem (no \*arvore){

if(arvore != NULL){

printf("%d\n",arvore->chave);

preordem(arvore->esq);

preordem(arvore->dir);

}

}

void emordem(no \*arvore){

if(arvore != NULL){

emordem(arvore->esq);

printf("%d\n",arvore->chave);

emordem(arvore->dir);

}

}

void posordem(no \*arvore){

if(arvore != NULL){

posordem(arvore->esq);

posordem(arvore->dir);

printf("%d\n",arvore->chave);

}

}

int menu(){

int op;

printf("\nMENU\n\n");

printf("1 - Inserir valor\n");

printf("2 - Imprimir pre ordem\n");

printf("3 - Imprimir in-ordem\n");

printf("4 - Imprimir pos ordem\n");

printf("5 - Consulta detalhada\n");

printf("6 - Quantidades de nos da arvore\n");

printf("7 - Imprimir arvore\n");

printf("0 - Sair....\n\n");

printf("Digite sua opcao: ");

scanf("%d", &op);

return op;

}

int main(){

int n,a,b;

no \*raiz, \*aux;

raiz = NULL;

int opcao;

while(opcao!=12){

opcao = menu();

switch(opcao){

case 1:

printf("\n>>>>>>>>>>>>><<<<<<<<<<<<<\n\n");

printf("Digite -1 para terminar\n");

printf("\n>>>>>>>>>>>>><<<<<<<<<<<<<\n\n");

do{

printf("Digite um numero: ");

scanf("%d", &n);

if(n!=-1){

insere(n, &raiz);

}

}while (n!=-1);

//imprime(raiz,0);

break;

case 2:

printf("\n>>>>>>>>>>>>><<<<<<<<<<<<<\n\n");

printf("------> Pre-Ordem <------\n");

printf("\n>>>>>>>>>>>>><<<<<<<<<<<<<\n\n");

preordem(raiz);

break;

case 3:

printf("\n>>>>>>>>>>>>><<<<<<<<<<<<<\n\n");

printf("------> Em-Ordem <------\n");

printf("\n>>>>>>>>>>>>><<<<<<<<<<<<<\n\n");

emordem(raiz);

break;

case 4:

printf("\n>>>>>>>>>>>>><<<<<<<<<<<<<\n\n");

printf("------> Pos-Ordem <------\n");

printf("\n>>>>>>>>>>>>><<<<<<<<<<<<<\n\n");

posordem(raiz);

break;

case 5:

consultarDetalhes(raiz);

break;

case 6:

b=contarNos(raiz);

printf("\n>>>>>>>>>>>>><<<<<<<<<<<<<\n\n");

printf(" A arvore possui %d nos!\n",b);

printf("\n>>>>>>>>>>>>><<<<<<<<<<<<<\n\n");

break;

case 7:

printf("ARVORE BINARIA\n");

imprime(raiz,0);

break;

default :

printf("opcao nao existe! tente novamente");

break;

case 0:

system("pause");

break;

}

}

return 0;

}