Universidade Federal Fluminense

Matéria Estrutura de Dados

Professor Dalessandro Soares

Alunos: Alessandro Sampaio e Tatiane Sousa

#include <stdlib.h>

#include <stdio.h>

#include <math.h>

typedef struct arvore{

int info;

struct arvore \*esq;

struct arvore \*dir;

}Arvore;

Arvore\* Ler\_arvore\_arq(FILE \*arq){

char c;

int x;

fscanf(arq,"%c",&c);

fscanf(arq,"%d",&x);

if(x==-1){

fscanf(arq,"%c",&c);

return NULL;

}

else{

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

no->info=x;

no->esq=Ler\_arvore\_arq(arq);

no->dir=Ler\_arvore\_arq(arq);

fscanf(arq,"%c",&c);

return no;

}

}

void imprimir\_preordem (Arvore\* a){

if(a!=NULL)

{

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

imprimir\_preordem(a->esq);

imprimir\_preordem(a->dir);

}

}

void imprimir\_ordem (Arvore\* a){

if(a!=NULL)

{

imprimir\_ordem(a->esq);

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

imprimir\_ordem(a->dir);

}

}

void imprimir\_posordem (Arvore\* a){

if(a!=NULL)

{

imprimir\_posordem(a->esq);

imprimir\_posordem(a->dir);

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

}

}

void imprimir\_nivel(Arvore\* a, int nivel, int cont){

if(a!=NULL){

if(cont==nivel)

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

else{

imprimir\_nivel(a->esq,nivel,cont+1);

imprimir\_nivel(a->dir,nivel,cont+1);

}

}

}

void imprimir\_largura(Arvore\* a, int nivel, int cont, int w){

if(a!=NULL){

if(nivel<w){

imprimir\_nivel(a,nivel,cont);

imprimir\_largura(a->esq,nivel+1,cont+1,w);

imprimir\_largura(a->dir,nivel+1,cont+1,w);

}

}

}

int imprimir\_nivel\_no(Arvore\* a, int y, int cont){

if(a==NULL)

return 0;

else{

if(a->info==y)

return cont;

else

return imprimir\_nivel\_no(a->esq,y,cont+1)+imprimir\_nivel\_no(a->dir,y,cont+1);

}

}

int altura(Arvore\* a){

if(a==NULL)

return 0;

else{

int ae,ad;

ae=altura(a->esq);

ad=altura(a->dir);

if(ae>ad)

return ae+1;

else

return ad+1;

}

}

int existe(Arvore \*a,int y){

if(a==NULL)

return 0;

else{

if(a->info==y)

return 1;

else{

if(existe(a->esq,y)==1)

return 1;

else

return existe(a->dir,y);

}

}

}

int numero\_nos(Arvore\* a){

if(a==NULL)

return 0;

else

return 1+numero\_nos(a->esq)+numero\_nos(a->dir);

}

Arvore\* altera\_elemento(Arvore\* a, int y){

if (a==NULL)

return a;

if(a->info==y){

int z;

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

scanf("%d",&z);

a->info=z;

return a;

}

else{

a->esq=altera\_elemento(a->esq,y);

a->dir=altera\_elemento(a->dir,y);

return a;

}

}

int conta\_nos(Arvore\* a, int nivel, int cont){

if(a==NULL)

return 0;

else{

if(cont==nivel)

return 1;

else

return conta\_nos(a->esq,nivel,cont+1)+conta\_nos(a->dir,nivel,cont+1);

}

}

int arvore\_cheia (Arvore\* a){

int w,n,cont=0;

w=altura(a);

n=conta\_nos(a,w-1,cont);

w=w-1;

cont= pow(2,w);

if(a!=NULL){

if(cont==n)

return 1;

else

return 0;

}

}

Arvore\* remover\_arvore(Arvore\* a){

if (a!=NULL){

remover\_arvore(a->esq);

remover\_arvore(a->dir);

free(a);

}

return NULL;

}

int main (void){

FILE\* arq;

arq=fopen("arvore.txt","rt");

if (arq==NULL){

printf ("erro ao abrir o arquivo");

exit (1);

}

int x=1,y=0,w=0,nivel=0,cont=0;

Arvore \*a=NULL;

while (x!=8){

system("cls");

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

printf ("1 - Ler a arvore de um arquivo\n");

printf ("2 - Opcoes de impressao da arvore\n");

printf ("3 - Calcular a altura da arvore\n");

printf ("4 - Verificar se um elemento existe na arvore\n");

printf ("5 - Contar o numero nos na arvore\n");

printf ("6 - Alterar um o conteudo de um no da arvore\n");

printf ("7 - Verificar se a arvore esta cheia\n");

printf ("8 - Sair\n");

scanf ("%d",&x);

system("cls");

switch (x)

{

case 1:{

a=Ler\_arvore\_arq(arq);

printf ("A arvore foi lida com sucesso\n");

system("pause");

break;

}

case 2:{

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

printf ("1 - Para imprimir em Pre-ordem\n");

printf ("2 - Para imprimir em Ordem\n");

printf ("3 - Para imprimir em Pos-ordem\n");

printf ("4 - Para imprimir em Largura\n");

printf ("5 - Para imprimir o nivel da arvore\n");

printf ("6 - Para imprimir o nivel de um determinado no\n");

scanf ("%d",&y);

system("cls");

switch (y){

case 1:{

printf ("Impressao em pre-ordem\n\n");

imprimir\_preordem(a);

printf (" \n");

system("pause");

break;

}

case 2:{

printf ("Impressao em ordem\n\n");

imprimir\_ordem(a);

printf (" \n");

system("pause");

break;

}

case 3:{

printf ("Impressao em pos-ordem\n\n");

imprimir\_posordem(a);

printf (" \n");

system("pause");

break;

}

case 4:{

printf ("Impressão em Largura\n\n");

nivel=0;

w=altura(a);

imprimir\_largura(a,nivel,cont,w);

printf (" \n");

system("pause");

break;

}

case 5:{

printf ("Impressão em Nivel\n\n");

printf ("Digite o nivel desejado\n");

scanf ("%d",&nivel);

printf (" \n");

w=altura(a);

if(nivel>(w-1)){

printf("A arvore nao contem esse nivel\n");

system("pause");

}

else{

printf("Os elementos do nivel %d: \n",nivel);

imprimir\_nivel(a,nivel,cont);

printf (" \n");

system("pause");

}

break;

}

case 6:{

printf ("Impressao do nivel de um determinado noh\n\n");

printf ("Digite o noh desejado\n");

scanf ("%d",&y);

w=existe(a,y);

if (w==0)

printf("Noh nao existente na arvore");

else{

cont=0;

w=imprimir\_nivel\_no(a,y,cont);

printf ("O nivel do noh %d eh %d\n",y,w);

}

printf (" \n");

system("pause");

break;

}

}

break;

}

case 3:{

w=altura(a);

printf("A altura da arvore eh %d\n", w);

system("pause");

break;

}

case 4:{

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

scanf ("%d",&y);

w=existe(a,y);

if (w==1)

printf ("Elemento existe\n");

else

printf ("Elemento nao existe\n");

system("pause");

break;

}

case 5:{

w=numero\_nos(a);

printf ("O numero de nos da arvore eh %d\n",w);

system("pause");

break;

}

case 6:{

printf ("Digite o elemento para ser alterado\n");

scanf ("%d",&y);

w=existe(a,y);

if (w==1){

printf ("Elemento encontrado\n");

a=altera\_elemento(a,y);

}

else

printf ("Elemento nao existe\n");

system("pause");

break;

}

case 7: {

w=arvore\_cheia(a);

if (w==0)

printf ("A arvore lida nao estah cheia\n");

else

printf ("A arvore lida esta cheia\n");

system ("pause");

break;

}

case 8: {

a=remover\_arvore(a);

break;

}

}

}

fclose(arq);

return 0;

}