UNIVERSIDADE ESTADUAL DE PONTA GROSSA

SETOR DE CIÊNCIAS AGRÁRIAS E SUAS TÉCNOLOGIAS

DEPARTAMENTO DE INFORMÁTICA

RENANN RODRIGUES DA SILVA

MAYSA LOVATTO LOPES

TRABALHO DE ESTRUTURA DE DADOS

PONTA GROSSA-PR

2014

RENANN RODRIGUES DA SILVA

MAYSA LOVATTO LOPES

TRABALHO DE ESTRUTURA DE DADOS

trabalho apresentado para

a obtenção de nota referente

ao 2° bimestre, na disciplina

de Estrutura de dados

PONTA GROSSA

2014

/\* Acadêmicos: RENANN R. DA SILVA RA:13106523

MAYSA LOVATTO LOPES RA:13015123

================================================================================

Trabalho 2°bi Estrutura de Dados

================================================================================

=>Programa deve ler palavras de um .txt

=>Ordená-las em uma arvore binária.

=>Imprimir em ordem alfabética contando as paginas e

quantas vezes a palavra apareceu

\*/

#include<stdio.h>

#include<conio.h>

#include<string.h>

#define TAM 50

struct no{

char palavra[50];

int contador;

int pag[TAM];

no \*dir;

no \*esq;

};

struct no \*raiz;

int pagina = 1;

**void insere**(char valor[]){

no \*aux;

no \*atual;

aux = new(struct no);

aux->esq = NULL;

aux->dir = NULL;

strcpy(aux->palavra,valor);

aux->contador = 1; //contador de ocorrência da palavra

aux->pag[0] = pagina; //paginas de ocorrência

if(raiz == NULL){

raiz = aux;

return;

}

atual = raiz;

while(1){

if(strcmpi(valor, atual->palavra) > 0){

if(atual->dir == NULL){

atual->dir = aux;

return;

}

atual = atual->dir;

}

if(strcmpi(valor, atual->palavra) < 0){

if(atual->esq == NULL){

atual->esq = aux;

return;

}

atual = atual->esq;

}

}

**}**

**void em\_ordem**(struct no\* atual){

int j=0;

if(atual!=NULL){

em\_ordem(atual->esq);

printf("%s -", atual->palavra); //imprime a palavra

printf(" %d vezes nas paginas = ", atual->contador); //imprime quantas vezes ela apareceu

while(j < atual->contador){

printf("%d,",atual->pag[j]); //imprime as paginas que a palavra apareceu

j++;

}

printf("\n");

em\_ordem(atual->dir);

}

**}**

**void verifica**(char valor[]){

/\* Esse procedimento verifica se a palavra passada ja existe na arvore e em qual

pagina esta no momento.

Caso a palavra já esteja na arvore, somente é atualizado os campos "contador","pag"

Se "void verifica" não encontrar a palavra é chamado o procedimento insere.

\*/

no \*atual;

if(strcmpi(valor, "[QUEBRA]") == 0 ){

pagina = pagina+1; //pagina++ cada vez que [QUEBRA] aparece no .txt

return;

}

if(raiz == NULL){

insere(valor);

return;

}

atual = raiz;

while(atual != NULL){

if(strcmpi(valor, atual->palavra) >= 0){

if(strcmpi(valor, atual->palavra) == 0){ //se palavra já existe

atual->contador++;//atualiza-se contador

atual->pag[atual->contador-1] = pagina;//vetor de pag recebe pag atual

return;

}

atual = atual->dir;

}

if(strcmpi(valor, atual->palavra) <= 0){

if(strcmpi(valor, atual->palavra) == 0 ){

atual->contador++;

atual->pag[atual->contador-1] = pagina;

return;

}

atual = atual->esq;

}

}

insere(valor);//caso a palavra não esteja na arvore ela é inserida na arvore

**}**

**main**(){

char palavra[TAM];

FILE \*fp;

fp = fopen("texto.txt", "r");

while(!feof(fp)){

fscanf(fp,"%s", palavra);

if(strcmpi(palavra, "") != 0)

verifica(palavra);

strcpy(palavra,"");//Pra nao correr o risco de inserir lixo de memória..

} // ..a variável palavra é "zerada"

em\_ordem(raiz);

getch();

**}**