Trabalho de Estrutura de Dados 2

Alunos:

João Vítor da Silva - 12011BSI223

Israel Lúcio De Lima Vaz - 12011BSI220

Pedro Milvar Santos Vieira – 11921BSI207

Link do GitHub: https://github.com/PedroMilvar/ED2.git

Problema:

Um índice remissivo de um livro, lista os termos e tópicos que são abordados num documento, em ordem alfabética, juntamente com as páginas em que aparecem no livro. Seguindo a ideia de criar um índice remissivo para um documento texto, considerando apenas os termos (palavras), cada termo encontrado no documento teria uma lista de linhas em que foi encontrado. Devem ser desconsideradas diferenças entre letras maiúsculas e minúsculas, e se a palavra aparece mais de uma vez numa determinada linha, esta linha aparece apenas uma vez no índice.

Exemplo: (Machado de Assis: A uma senhora que me pediu versos)

Os sóis e as luas Creio bem que Deus os fez Para outras vidas

---

Este projeto envolve a criação de um índice remissivo para um documento utilizando árvore binária AVL. Cada nó da árvore AVL deve conter uma palavra e um vetor para armazenar as linhas em que aquela palavra aparece no documento. Apenas as funções de busca e inserção serão utilizadas neste projeto. Não haverá remoção de palavras do índice e pontuações poderão ser utilizadas como delimitadores de palavras, mas não constam do índice. O índice deve ser retornado em um arquivo texto (.txt) com as três linhas finais (total de palavras, total de palavras distintas, tempo de construção), e o documento de entrada também deve ser do tipo texto simples.

Estrutura:

Para a realização do processo pedido foi utilizado as seguintes estruturas:

Árvore AVL:

- Árvore AVL é uma árvore binária de busca balanceada, ou seja, uma árvore balanceada são as árvores que minimizam o número de comparações efetuadas no pior caso para uma busca com chaves de probabilidades de ocorrências idênticas;

TAD (Tipo Abstrato de Dados):

- Encapsulamento: Encapsular os dados de uma aplicação significa evitar que estes sofram acessos indevidos;
- Segurança:Dificulta a violação de memória ;
- Flexibilidade:Pode-se modificar o código do TAD sem que seu cabeçalho (interface) seja alterado ;
- Reutilização:O uso de módulos facilita o compartilhamento para diferentes programas e usuários;

Principais Funções:

insere_ArvAVL: Inserção e criação dos NÓS;

```
int insere_ArvAVL(ArvAVL *raiz, char* palavra, int linha){
    int r;
    if(*raiz == NULL){//árvore vazia ou nó folha
        struct NO *novo;
        novo = (struct NO*)malloc(sizeof(struct NO));
        if(novo == NULL)
            return 0;

        strcpy(novo->palavra, palavra);
        novo->altura = 0;
        novo->esq = NULL;
        novo->dir = NULL;
        novo->linha[linha] = 1;
        *raiz = novo;
        return 1;
    }
```

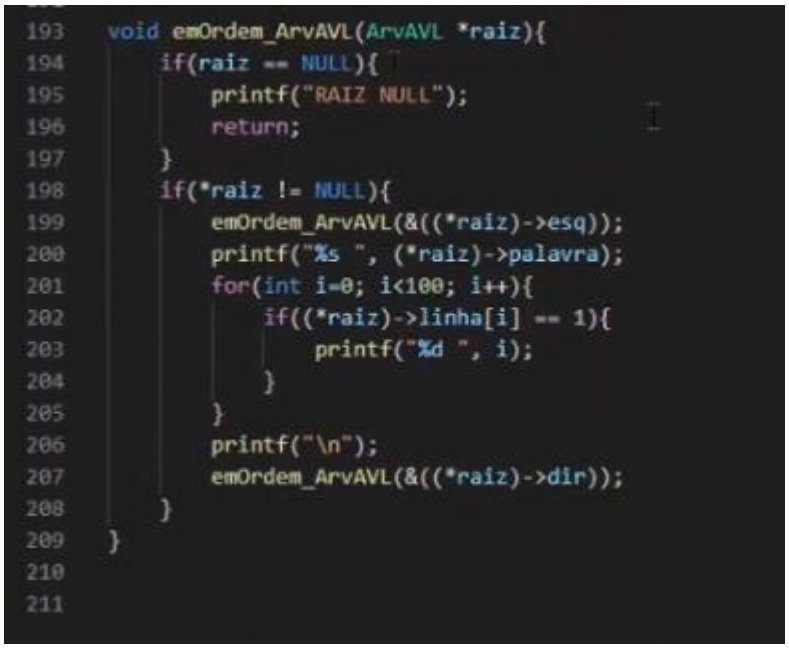
```
162     struct NO *atual = *raiz;
163     if(strcmp(palavra, atual->palavra) < 0){
164         r = insere_ArvAVL(&(atual->esq), palavra, linha);
165         if(r == 1){
166             if(fatorBalanceamento_NO(atual) >= 2){
167                 if(strcmp(palavra, (*raiz)->esq->palavra) < 0){
168                     RotacaoDireita(raiz);
169                 }else{
170                     RotacaoDuplaDireita(raiz);
171                 }
172             }
173         }
174     }else{
175         if(strcmp(palavra,atual->palavra) > 0){
176             r = insere_ArvAVL(&(atual->dir), palavra, linha);
177             if(r == 1){
178                 if(fatorBalanceamento_NO(atual) >= 2){
179                     if(strcmp((*raiz)->dir->palavra, palavra) < 0){
180                         RotacaoEsquerda(raiz);
181                     }else{
182                         RotacaoDuplaEsquerda(raiz);
183                     }
184                 }
185             }
186         }else{
187             printf("Valor duplicado!!\n");
188             return 0;
189         }
190     }
191     atual->altura = maior(altura_NO(atual->esq),altura_NO(atual->dir)) + 1;
192
193     return r;
```

opentxt: Abertura do arquivo .txt, entrada dos dados do txt.;

cria_ArvAVL: Criação da árvore;

```
16  void opentxt(char *arquivo, char *vetaux){
17          FILE *fp = fopen(arquivo, "r");
18          if(fp == NULL){
19              return ;
20          }
21          char c;
22          int i=0;
23          c = fgetc(fp);
24          while(c != EOF){
25              vetaux[i] = c;
26              c = fgetc(fp);
27              i++;
28          }
29  }
30
31  ArvAVL* cria_ArvAVL(){
32      ArvAVL* raiz = (ArvAVL*) malloc (sizeof(ArvAVL));
33      if(raiz != NULL)
34          *raiz = NULL;
35      return raiz;
36  }
```

emOrdem_ArvAVL: Mostrar a árvore criada pela função anterior;



```
193  void emOrdem_ArvAVL(ArvAVL *raiz){
194      if(raiz == NULL){
195          printf("RAIZ NULL");
196          return;
197      }
198      if(*raiz != NULL){
199          emOrdem_ArvAVL(&((*raiz)->esq));
200          printf("%s ", (*raiz)->palavra);
201          for(int i=0; i<100; i++){
202              if((*raiz)->linha[i] == 1){
203                  printf("%d ", i);
204              }
205          }
206          printf("\n");
207          emOrdem_ArvAVL(&((*raiz)->dir));
208      }
209  }
210
211
```

Exemplos rodados:

Os sóis e as luas

Creio bem que Deus os fez

Para outras vidas