TRABALHO FINAL
Contador de Frequências de Palavras

1 Objetivo

Várias tarefas nas áreas de recuperação de informações e processamento de linguagem natural necessitam calcular a frequência de ocorrência das palavras em um texto. Por exemplo, em uma *tag cloud,* as palavras maiores serão aquelas com maior frequência. Além disso, motores de busca utilizam vários indicadores para montar o ranking das páginas mostradas em resposta a uma consulta, dentre eles, a frequência de ocorrência das palavras da consulta no texto da página.

Sendo assim, a aplicação a ser desenvolvida como trabalho final da disciplina é ***um contador de frequências de palavras em um texto***. Uma **palavra** é uma sequência de letras. Todos os outros caracteres (espaço, pontuação, etc.) deverão ser considerados como separadores de palavras. Diferenças entre letras maiúsculas e minúsculas devem ser desprezadas (ex: a = A).

Dados de Entrada:

- arquivo com o texto a ser analisado e
- arquivo com as operações a serem realizadas.

Dados de Saída:

- arquivo texto contendo:
  - estatísticas da árvore gerada incluindo: número de nodos, altura, fator de balanceamento, tempo decorrido, número de rotações e de comparações necessárias para processar o texto de entrada e carregá-lo na estrutura (para listas, apenas o tempo decorrido, número de nodos e o de comparações precisa ser impresso) e
  - resultado de cada uma das operações realizadas juntamente com o tempo e o número de comparações necessárias para gerar a saída.

Funções necessárias:

- Frequência (**F**) – dada uma palavra, retornar seu número de ocorrências.

```
int frequencia (char * palavra)
```

- Contador (**C**) – imprime as palavras que têm n>= k_1 e n <= k_2 em ordem decrescente de frequência de ocorrência. Palavras com a mesma frequência, devem ser listadas em ordem alfabética.

```
void contador (int k1, int k2)
```

Funções auxiliares

- **ContaNodos** – retorna o número de nodos da árvore (serve para checar se todos os números foram corretamente inseridos/removidos).
- **Altura** – retorna a altura da árvore (apenas para árvores)
- **Fator** – retorna o fator de balanceamento da árvore (apenas para árvores).

É preciso também contar o **número de comparações** (*i.e.*, o número de vezes que um valor de chave é comparado com o valor do nodo corrente **e rotações** (número de vezes que uma operação de rotação é chamada) efetuadas para processar o total de valores em um arquivo.

2. Requisitos

- Fazer **duas** versões da aplicação, uma delas deve obrigatoriamente utilizar árvores balanceadas (AVL, Rubro-Negras ou Splay) e a outra pode utilizar qualquer uma das estruturas (árvores ou listas) trabalhadas na disciplina. Exemplos de comparativos possíveis: AVL vs ABP, Splay vs LSE, AVL vs Rubro Negra, etc. A estrutura deve ser utilizada para **armazenar cada palavra do texto** juntamente com a sua frequência.

- Fazer **um comparativo** entre as duas versões e descrevê-lo **detalhadamente** no relatório. Este comparativo deve utilizar como base o número de

- A aplicação deve ser chamada **a partir da linha de comando** (passando parâmetros para o main). Por exemplo, o comando

```
C:\contador texto1.txt operacoes.txt saida.txt
```

  significa que é necessário processar o texto de nome `texto1.txt`, aplicar as operações especificadas no arquivo `operacoes.txt` e gravar a saída no arquivo `saida.txt`.

- O trabalho deve ser feito, preferencialmente, em duplas. A linguagem de programação aceita é C (Não é C++ nem C#).

3. Data de Entrega

- **04/12/19** apresentação (horário da aula) e entrega pelo Moodle

4. Critérios de Avaliação

O trabalho deve ser realizado em duplas e deverá ser apresentado e defendido na data prevista.
Para a avaliação serão adotados os seguintes critérios:

- relatório comparativo entre as duas implementações (35%);
- funcionamento (30%) e
- organização e documentação do código (35%).

Importante:

Este trabalho deverá representar a solução da dupla para o problema proposto. O plágio é terminantemente proibido e a sua detecção incorrerá na divisão da nota obtida pelo número de alunos envolvidos. Para detectar o plágio, usaremos o software MOSS (http://theory.stanford.edu/~aiken/moss/).

Exemplo de funcionamento

Entrada: `teste.txt`

```
A Lua encontra-se em rotação sincronizada com a Terra, mostrando sempre a mesma face visível, marcada por mares vulcânicos escuros entre montanhas cristalinas e proeminentes crateras de impacto. É o mais brilhante objeto no céu a seguir ao Sol, embora a sua superfície seja na realidade escura, com uma refletância pouco acima da do asfalto. A sua proeminência no céu e o seu ciclo regular de fases tornaram a Lua, desde a antiguidade, uma importante referência cultural na língua, em calendários, na arte e na mitologia. A influência da gravidade da Lua está na origem das marés oceânicas e ao aumento do dia sideral da Terra. A sua atual distância orbital, cerca de trinta vezes o diâmetro da Terra, faz com que no céu o satélite pareça ter o mesmo tamanho do Sol, permitindo-lhe cobri-lo por completo durante um eclipse solar total.
```

Entrada: `operacoes.txt`

```
F lua
F a
F bola
C 2 10
```

Comando: `C:\contador teste.txt operacoes.txt saida.txt`

Saída: `saida.txt`

```
**********ESTATÍSTICAS DA GERAÇÃO DA ÁRVORE AVL *************
Número de nodos: 100
Altura: 8
Fator de Balanceamento: 1
Tempo: 0.00000 ms
Rotações: 72
Comparações 2678
************************************************************
F lua
lua: 3 ocorrências
*************************
F a
a: 10 ocorrências
*************************
F bola
bola: 0 ocorrências
*************************
C 2 10
a 10
da 5
na 5
o 5
e 4
com 3
céu 3
de 3
do 3
lua 3
no 3
sua 3
terra 3
ao 2
em 2
por 2
sol 2
uma 2

Tempo: 1.00000 ms
Comparações 197
```