SI400 – Programação Orientada a Objetos II

Prof. André F. de Angelis
Oferecimento 2S2023

Projeto I: Pré analisador de textos

Objetivo: desenvolver um programa em **Java** para realizar a etapa de pré-processamento de um texto a ser analisado como dígrafo pela ferramenta de software livre Gephi. O programa deverá aceitar em linha de comando uma lista de nomes de arquivos do tipo texto simples (.txt) e, para cada um deles, gravar uma saída no formato .csv contendo listas de adjacências indicando sucessão de palavras para posterior construção dos dígrafos, como descrito a serguir:

- cada palavra é um nó único do dígrafo (início de uma lista de adjacência);
- o nó A tem um arco para o nó B se e somente se a palavra representada por B segue aquela representada por A no texto;
- cada linha do arquivo de saída inicia-se com a palavra que representa o nó de origem, seguida das palavras que são os destinos dos arcos que partem dela, separadas por espaços ou vírgulas (formato Gephi).

Perceba que toda a pontuação, aspas, travessões e tais devem ser eliminados, enquanto que todo o texto deve ser convertido para minúsculas. Cada palavra do texto aparece uma e uma só vez como a palavra inicial das linhas, em ordem alfabética. A lista dos destinos pode ter uma ou mais palavras, mas sem repetição na linha.

Veja o exemplo:

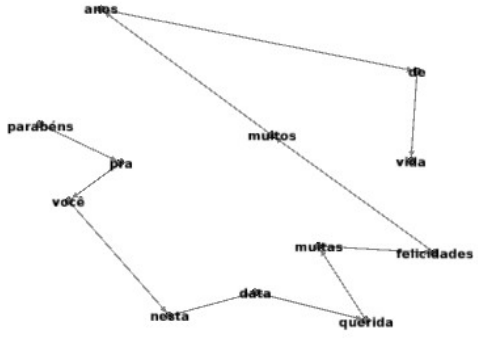
Entrada:

Parabéns pra você
Nesta data querida
Muitas felicidades
Muitos anos de vida

Saída:

anos, de
data, querida
de, vida
felicidades, muitos
muitas, felicidades
muitos, anos
nesta, data
parabéns, pra
pra, você
querida, muitas
você, nesta

Dígrafo:



Os seus arquivo de saída serão submetidos ao Gephi para revelar o dígrafo correspondente as sucessões de palavras. Veja o exemplo a seguir, com trecho da canção “Luar do Sertão”.

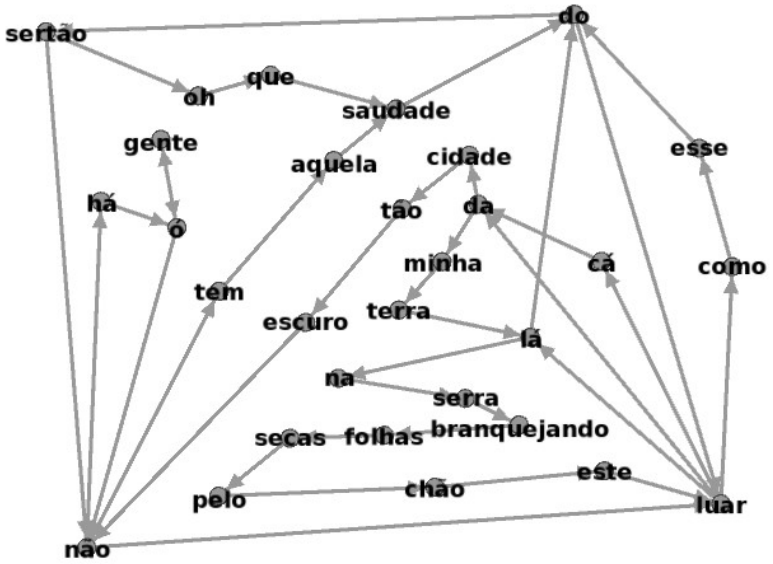
Entrada:

Não há, ó gente, ó não
Luar como esse do sertão
Não há, ó gente, ó não
Luar como esse do sertão
Oh que saudade do luar da minha terra
Lá na serra branquejando folhas secas pelo chão
Este luar cá da cidade tão escuro
Não tem aquela saudade do luar lá do sertão
Não há, ó gente, ó não
Luar como esse do sertão
Não há, ó gente, ó não
Luar como esse do sertão

Saída:

aquela, saudade
branquejando, folhas
chão, este
cidade, tão
como, esse
cá, da
da, cidade, minha
do, sertão, luar
escuro, não
esse, do
este, luar
folhas, secas
gente, ó
há, ó
luar, como, lá, da, cá
lá, na, do
minha, terra
na, serra
não, luar, há, tem
oh, que
pelo, chão
que, saudade
saudade, do
secas, pelo
serra, branquejando
sertão, oh, não
tem, aquela
terra, lá
tão, escuro
ó, gente, não

Dígrafo:



O programa deve ser constituído de, **ao menos,** as seguintes classes:

1. **AnalyzerStart**: partida do programa, leitura dos nomes de arquivos (via **args**[]), criação de objeto de controle (passe os arquivos validados como parâmetros para este construtor) e tratamento geral de erros;
2. **AnalyzerController**: orquestração das atividades das demais classes e tratamentos de exceções;
3. **AnalyzerReader**: leitura e *tokenização* de um arquivo de entrada, carga de uma estrutura de dados que represente internamente o dígrafo gerado a partir do texto e tratamentos de exceções;
4. **AnalyzerWriter**: escrita em um arquivo do dígrafo carregado pela classe anterior em uma estrutura de dados e tratamento de exceções;

Use mais classes se desejar!!!

Podem ser úteis as seguintes classes e interfaces da biblioteca do Java:

- ArrayList
- Map
- TreeMap

Implemente e depure o programa, gere a documentação com a ferramenta *javadoc*. O professor fornecerá exemplos de textos de entrada e de saída,para que você possa testar seu programa, e dois textos de validação. Submeta os arquivos de validação ao seu programa e, com as saídas produzidas, gere os correspondentes dígrafos no gephi, cuidando para que sejam legíveis.

A entrega do projeto será feita no Moodle com a postagem de **UM único** arquivo **pdf** contendo, *na ordem*:

- relatório de contribuição comentado (veja modelo no Moodle);
- todos os códigos-fonte, listados em sequência (por favor, separe os códigos de diferentes arquivos com algum delimitador visível na diagramação ou coloque cada um em página nova);
- figuras legíveis dos 2 grafos de validação.
- figuras legíveis mostrando execução do programa (não exagere! - dois ou três *prints* de tela são suficientes).

Se o grupo utilizou uma metodologia de desenvolvimento padronizada (*XP, Scum, Rup*, …), descreva no texto a aplicação dela no projeto (não é para dizer o que é ou como funciona, mas indicar como ela foi usada).

Critérios de Avaliação

- Só serão considerados os programas que compilem sem erro e executem corretamente a(s) funcionalidade(s) requerida(s).

#	Item	Max. pontos
		Pontuação Máxima
01	Entrega completa de todos os ítens solicitados	0,5
02	**Javadoc** para todas as classes e todos os métodos	1,0
03	Bônus para código e documentação totalmente em inglês	0,5
04	Arquitetura de classes (estruturação, divisão de responsabilidades)	2,0
05	Qualidade e consistência do código (estilo, indentação, organização, limpeza, nomeação)	2,0
06	Uso de estruturas de dados apropriadas	1,0
07	Tratamento de exceções adequado	0,5
08	Correção dos dígrafos gerados pelo Gephi	0,5
09	Bônus para metodologia de desenvolvimento padronizada	0,5
10	Aspecto geral do trabalho	1,5
	Total	10,0