UNIVERSIDADE FEDERAL DO ESPÍRITO SANTOS - UFES
ENGENHARIA DA COMPUTAÇÃO

Nathan Garcia Freitas

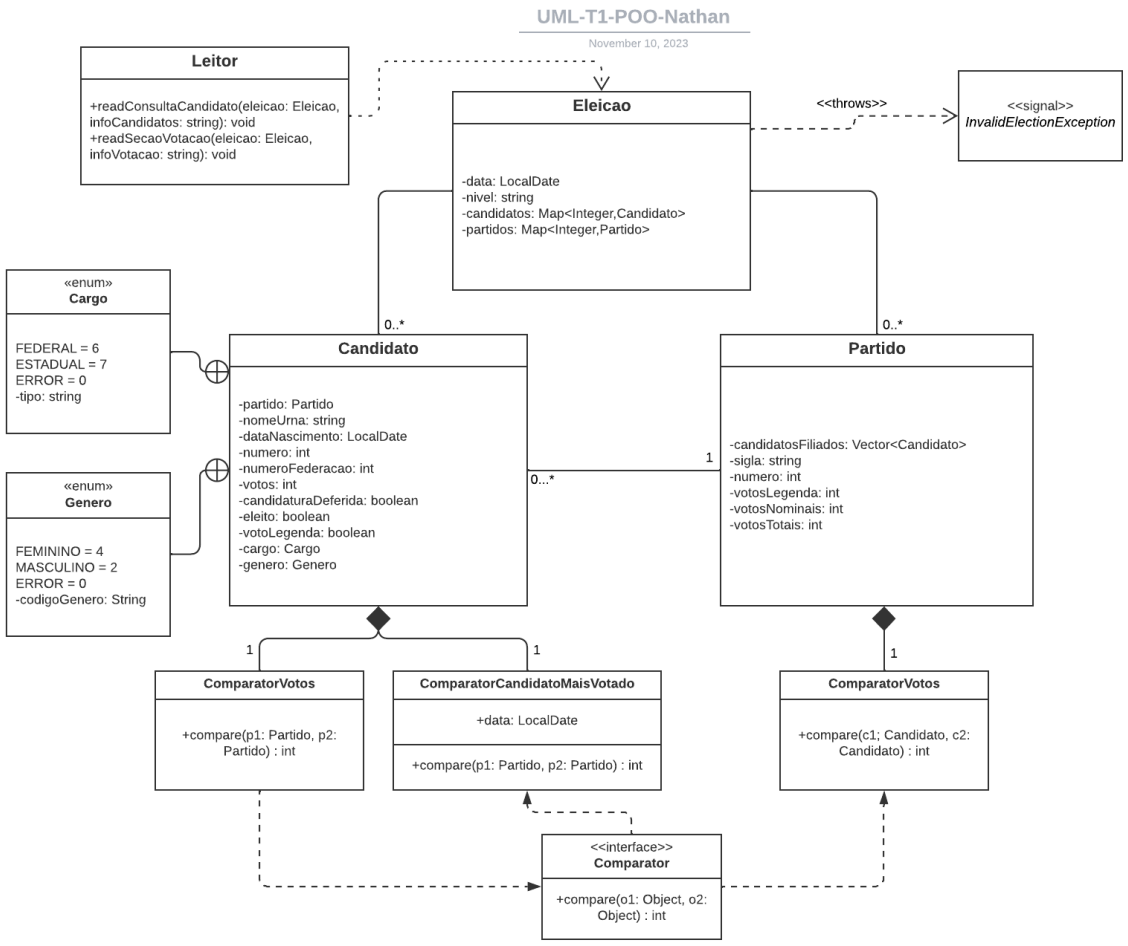
SISTEMA ELEITORAL BRASILEIRO ADAPTADO PARA PROGRAMAÇÃO ORIENTADAS A OBJETOS

VITÓRIA/ES
2023

1- DESCRIÇÃO DA IMPLEMENTAÇÃO

As classes utilizadas foram: Candidato, Eleição, Leitor e Partido, também contando com os seguintes enums: Gênero e Cargo, e por fim, a classe principal Main. Iniciando pela classe Main, ela é responsável por criar uma Eleição e um Leitor, ler os dados enviados pela linha de comando, e por fim, invocar a criação do relatório para a classe Eleição.

A imagem a seguir é referente ao diagrama de classe UML do projeto:



A classe Eleição, junto com a classe Leitor, são as responsáveis por ler, criar e armazenar todos os candidatos e todos os partidos inscritos para a respectiva eleição. A classe Leitor possui apenas dois métodos, sendo um deles para ler o arquivo que contém todas as informações de um Candidato e seu Partido, e o outro para ler o arquivo contendo informação sobre as votações. A classe Eleição junta todos os dados lidos pelo Leitor e armazena, com isso, gera um relatório com diversos dados requisitados na especificação do trabalho.

Tratando agora das classes de Candidato e Partido, elas são o cerne da Eleição. A classe Candidato possui atributos que dizem respeito a diversas informações da pessoa que irá disputar a candidatura, e também o partido em que ela é afiliada, possuindo um atributo Partido, e também, possuindo dois enum, sendo eles representando o Cargo que esse candidato irá disputar (Federal, Estadual ou Erro, caso seja outro tipo de cargo) e Gênero (Masculino, Feminino ou Erro, caso seja outro gênero). Já na classe Partido, possui atributos sobre esse partido, incluindo uma lista com todos os candidatos filiados ao partido.

Um detalhe a ressaltar é que ambas classes Partido e Candidato possuem classes implementadas da interface Comparator, em que a classe Candidato possui a classe ComparatorVotos, onde compara votos entre os candidatos e sendo a idade dos candidatos como critério de desempate, e a classe Partido possui também uma classe com o nome ComparatorVotos, em que compara os votos de cada partido (votos de legenda) e utiliza o partido com menor número como critério de desempate, e sua outra classe que implementa o Comparator, chama-se ComparatorCandidatoMaisVotado, que ao ser criado armazena a data da Eleição e compara os votos do candidato mais votado do partido, utilizando o critério de idade como desempate.

Lidando agora com as exceções, foram tratadas exceções envolvendo as operações de *parse,* como *LocalDate.parse* e *Integer.parseInt,* tratando as possíveis exceções lançadas por essas operações com o bloco try_catch, onde nos parâmetros passados pelo usuário, caso a data da eleição seja inválida, o programa encerrará. Também foi criado uma nova exceção nomeada como *InvalidElectionException,* onde caso o usuário passe na linha de comando um tipo de eleição diferente de “--federal” ou “--estadual”, será lançada essa exceção, encerrando o programa em sequência.

2- DESCRIÇÃO DOS TESTES

Os testes realizados contemplam os exemplos recomendados no *script* teste e também outros exemplos. Os estados utilizados nos testes são: AL, BA, ES, MG, PE, PI, RS, RO, SE e TO; sendo os seguintes estados fora dos testes apresentados pelo professor para executar o script: BA, ES, PI, RO, SE e TO. Com isso, foi utilizado o site https://resultados.tse.jus.br/oficial/app/index.html#/divulga para

comparar as respostas e também foram realizadas análises das informações do relatório que não estão explícitas no site (há de exemplo, eleitos por faixa etária, eleitos por gênero etc.).

3- BUGS

Após tratamento de qualquer possível erro no pipeline (data de eleição inválida, tipo de eleição inválida, caminho para os arquivos incorretos) e erros nos arquivos de candidato e votação, não foram encontrados mais bugs no programa.