Relatório Part.2

Engenharia de Software II

Gabriel Pereira 8150115

Nuno Josefino 8150189

Rafael Vieira 8150448



# Índice

[Índice 2](#_Toc535201261)

[Introdução 5](#_Toc535201262)

[Âmbito do Projeto 6](#_Toc535201263)

[No âmbito 6](#_Toc535201264)

[Fora do âmbito 6](#_Toc535201265)

[Diagrama de Use Cases 7](#_Toc535201266)

[Tecnologias Implementadas 7](#_Toc535201267)

[Project Management 11](#_Toc535201268)

[Project Development 12](#_Toc535201269)

[Estudo e ánalise dos Métodos 13](#_Toc535201270)

[Metódo isFound 13](#_Toc535201271)

[Tabela ECP 13](#_Toc535201272)

[Tabela BVA 13](#_Toc535201273)

[Casos de teste 13](#_Toc535201274)

[Caso de teste 1.1 () 13](#_Toc535201275)

[Caso de teste 1.2 () 13](#_Toc535201276)

[Metódo remove 14](#_Toc535201277)

[Tabela ECP 14](#_Toc535201278)

[Tabela BVA 14](#_Toc535201279)

[Casos de teste 14](#_Toc535201280)

[Caso de teste 2.1 () 14](#_Toc535201281)

[Caso de teste 2.2 () 14](#_Toc535201282)

[Metódo TratamentoPalavras 15](#_Toc535201283)

[Tabela ECP 15](#_Toc535201284)

[Tabela BVA 15](#_Toc535201285)

[Caso de teste 15](#_Toc535201286)

[Caso de teste 3.1 () 15](#_Toc535201287)

[Caso de teste 3.2 () 15](#_Toc535201288)

[Metódo countforQ 16](#_Toc535201289)

[Tabela ECP 16](#_Toc535201290)

[Tabela BVA 16](#_Toc535201291)

[16](#_Toc535201292)

[Caso de teste 16](#_Toc535201293)

[Caso de teste 4.1 () 16](#_Toc535201294)

[Caso de teste 4.2 () 16](#_Toc535201296)

[Metódo matrizQ 17](#_Toc535201298)

[Tabela ECP 17](#_Toc535201299)

[Tabela BVA 17](#_Toc535201300)

[Caso de teste 17](#_Toc535201301)

[Caso de teste 5.1 () 17](#_Toc535201302)

[Caso de teste 5.2 () 17](#_Toc535201303)

[Case de teste 5.3 () 17](#_Toc535201304)

[Caso de teste 5.4 () 17](#_Toc535201305)

[Metódo grauSimilariedade 18](#_Toc535201306)

[Tabela ECP 18](#_Toc535201307)

[Tabela BVA 18](#_Toc535201308)

[Caso de teste 18](#_Toc535201309)

[Caso de teste 6.1 () 18](#_Toc535201310)

[Caso de teste 6.2 () 18](#_Toc535201311)

[Caso de teste 6.3 () 18](#_Toc535201312)

[Metódo ordenado 19](#_Toc535201313)

[Tabela ECP 19](#_Toc535201314)

[Tabela BVA 19](#_Toc535201315)

[Caso de teste 19](#_Toc535201316)

[Caso de teste 7.1 () 19](#_Toc535201317)

[Caso de teste 7.2 () 19](#_Toc535201318)

[Metódo grauAcima 20](#_Toc535201319)

[Tabela ECP 20](#_Toc535201320)

[Tabela BVA 20](#_Toc535201321)

[Caso de teste 20](#_Toc535201322)

[Caso de teste 8.1 () 20](#_Toc535201323)

[Caso de teste 8.2 () 20](#_Toc535201324)

[Caso de teste 8.3 () 20](#_Toc535201325)

[Metódo maximoFicheiros 21](#_Toc535201326)

[Tabela ECP 21](#_Toc535201327)

[Tabela BVA 21](#_Toc535201328)

[Caso de teste 21](#_Toc535201329)

[Caso de teste 9.1 () 21](#_Toc535201330)

[Caso de teste 9.2 () 21](#_Toc535201331)

[Caso de teste 9.3 () 21](#_Toc535201332)

[Conclusão 22](#_Toc535201333)

[Anexos 23](#_Toc535201334)

[Github 23](#_Toc535201335)

[Máquina Virtual 23](#_Toc535201336)

# Introdução

O presente trabalho surge no âmbito da Unidade Curricular de Engenharia de Software II e em especifico da parte pratica da mesma.

Foi-nos apresentado um desafio no dominio da engenharia de software e em particular na áre da qualidade de software.

Desafio este que consistiu em desenvolver um motor de pesquisa que através da introdução de uma frase na query, infere sobre um repositório de ficheiros de texto, tendo sido necessário o desenvolvimento de uma bibilioteca de software capaz de processar coleções de ficheiros de modo a encontrar ficheiro com um grau de similariedade elevado através de uma query de pesquisa.

Realizamos um estudo prévio das restantes ferramentas que foram utilizadas por nós durante o decorrer do projeto.

Como último objetivo definimos a elaboração de um relatório de testes.

# Âmbito do Projeto

## No âmbito

No âmbito desta fase final do trabalho incluimos a definição de requisitos, realização de tabelas *ECP*, *BVA*, diagrama de casos de uso, codificação dos métodos e criação dos testes e elaboração dos relatório de testes.

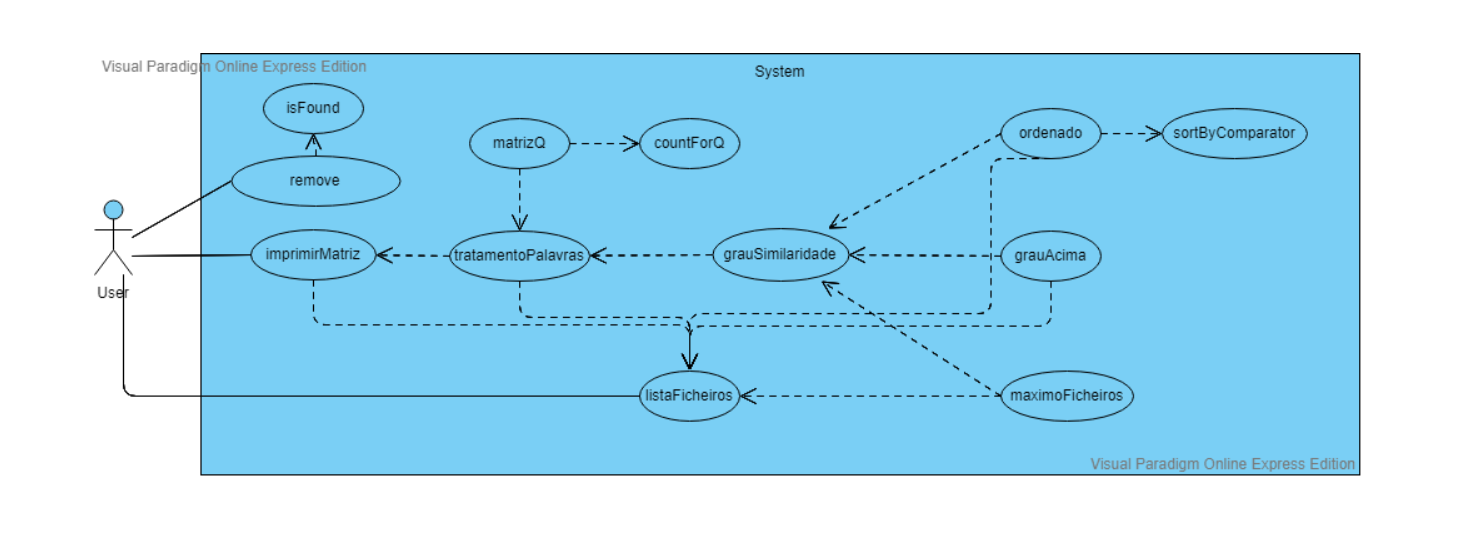
De igual forma, incluimos a criação de um repositório *GIT* e a sua configuração, o estudo e criação de um projeto no *Youtrack* enquanto ferramenta de *issue tracking* para o desenvolvimento do projeto.

É incluído também o estudo das restantes ferramentas que irão ser configuradas e utilizadas nos sprints seguintes deste projeto.

## Fora do âmbito

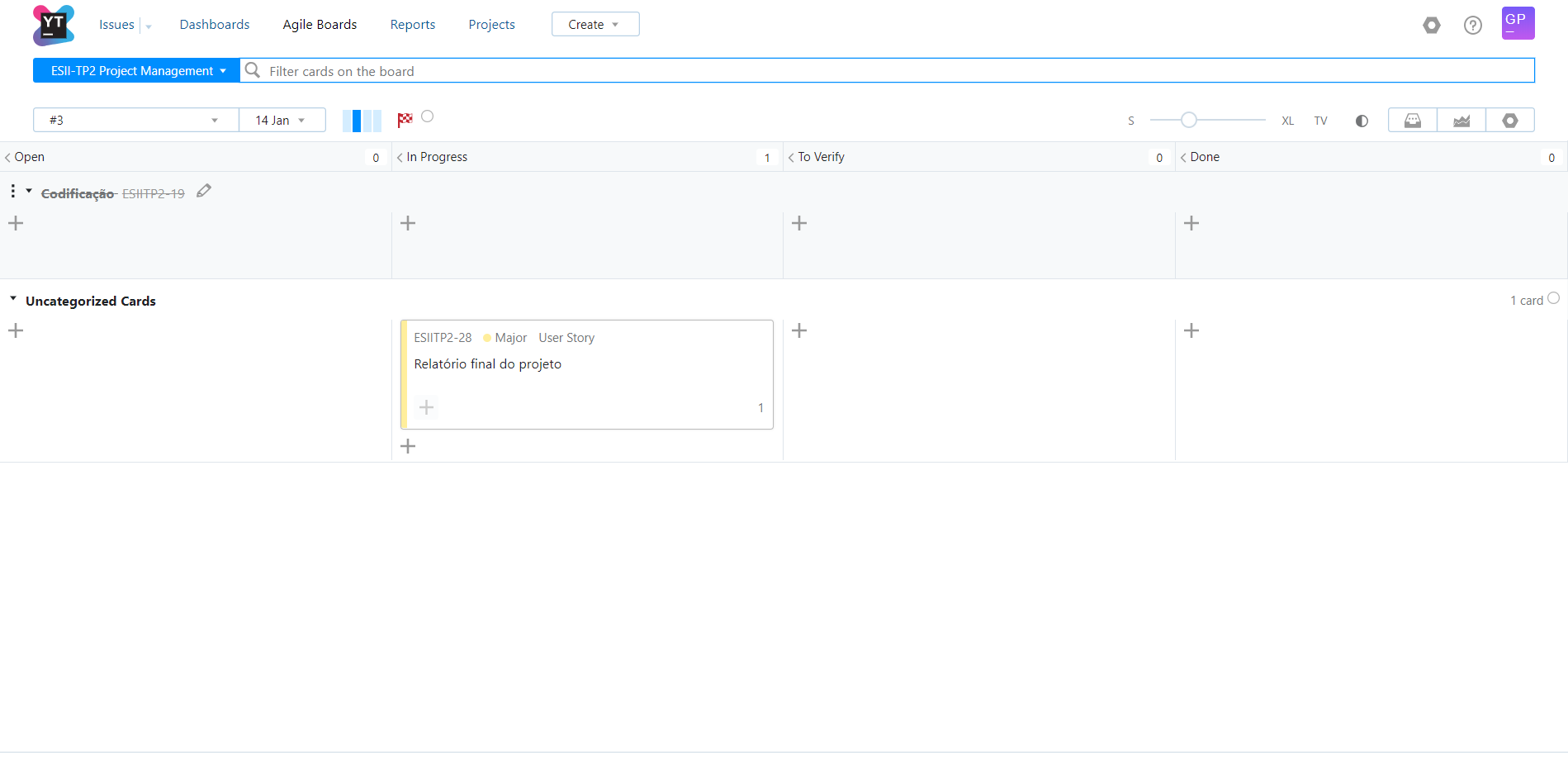
Fora do âmbito desta fase do projeto, incluimos o desenvolvimento de software funcional bem como a utilização de ferramentas para codificação de testes, revisão de código, ferramentas para implementação dos conceitos de *CI* durante o desenvolvimento do projeto como *builds* e testes automáticos ou linguagens de programação usada para o desenvolvimento do motor de pesquisa e especificação de testes.

# Diagrama de Use Cases

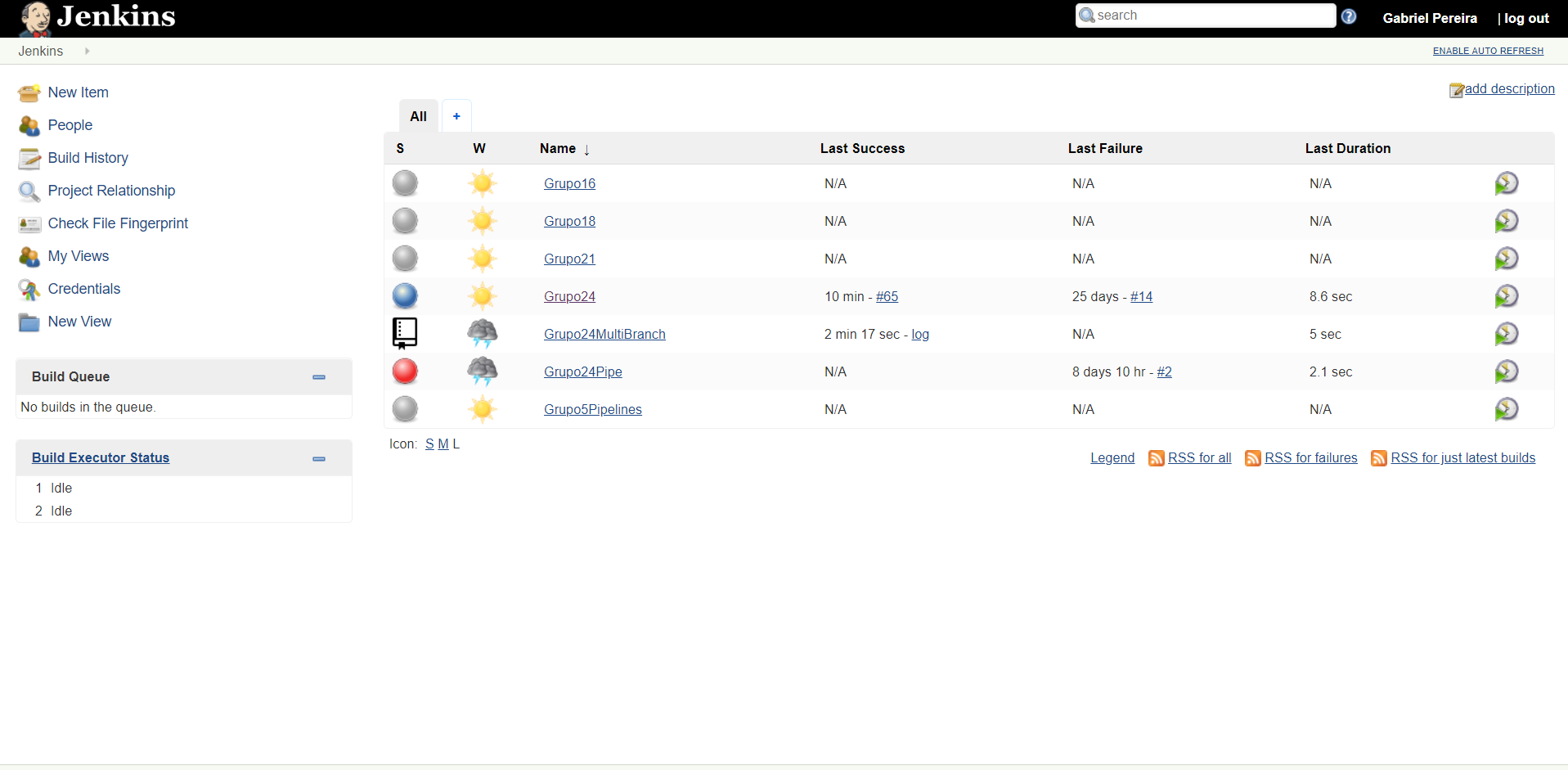


# Tecnologias Implementadas

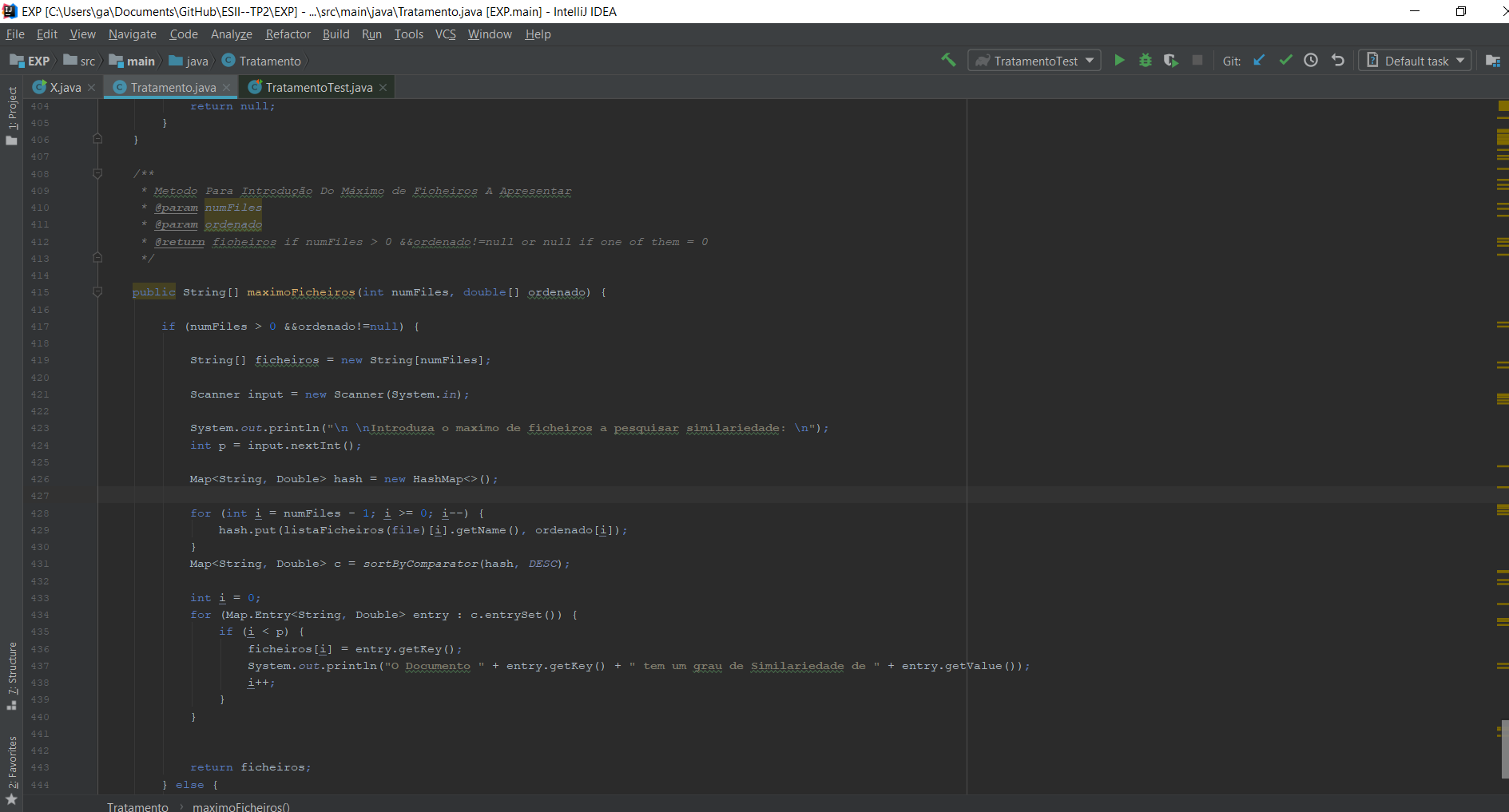
-O YouTrack é uma ferramenta desenvolvida pela **JetBrains**. É um Issue Tracker, mas tem, como de costume neste tipo de projeto, uma boa ponta de gerenciador de projeto agile. O seu principal destaque do **YouTrack,** conseguimos usá-lo em ilimitados projetos desde que a instalação padrão tenha até 10 utilizadores.



-O Jenkins é um servidor de automação de código aberto escrito em Java. É um sistema baseado em servidor que é executado em contêineres de servlet, como o Apache Tomcat. Suporta ferramentas de controle de versão, incluindo AccuRev, CVS, Subversion, Git, Mercurial, Perforce, TD / OMS, ClearCase e RTC, e pode executar projetos baseados em Apache Ant, Apache Maven e sbt, bem como scripts de shell arbitrários e comandos de lote do Windows. Com ele implementamos conceitos de Continuous Integration como builds e testes automáticos;



-IntelliJ IDEA é um ambiente de desenvolvimento integrado (IDE) Java para desenvolvimento de software de computador. Ele é desenvolvido pela JetBrains e está disponível como uma edição da comunidade licenciada do Apache 2, e em uma edição comercial proprietária. Ambos podem ser usados ​​para desenvolvimento comercial.



-Java é uma linguagem de programação. Em 2008 o Java foi adquirido pela empresa Oracle Corporation. Diferenças de linguagens de programação, que são compiladas para o código nativo, uma linguagem Java Java é compilada para um bytecode que é interpretada por uma máquina virtual (Java Virtual Machine, mais conhecida pela sua abreviação em JVM). Uma linguagem de programação Java é uma linguagem convencional da Plataforma Java, mas não é a sua linguagem única.



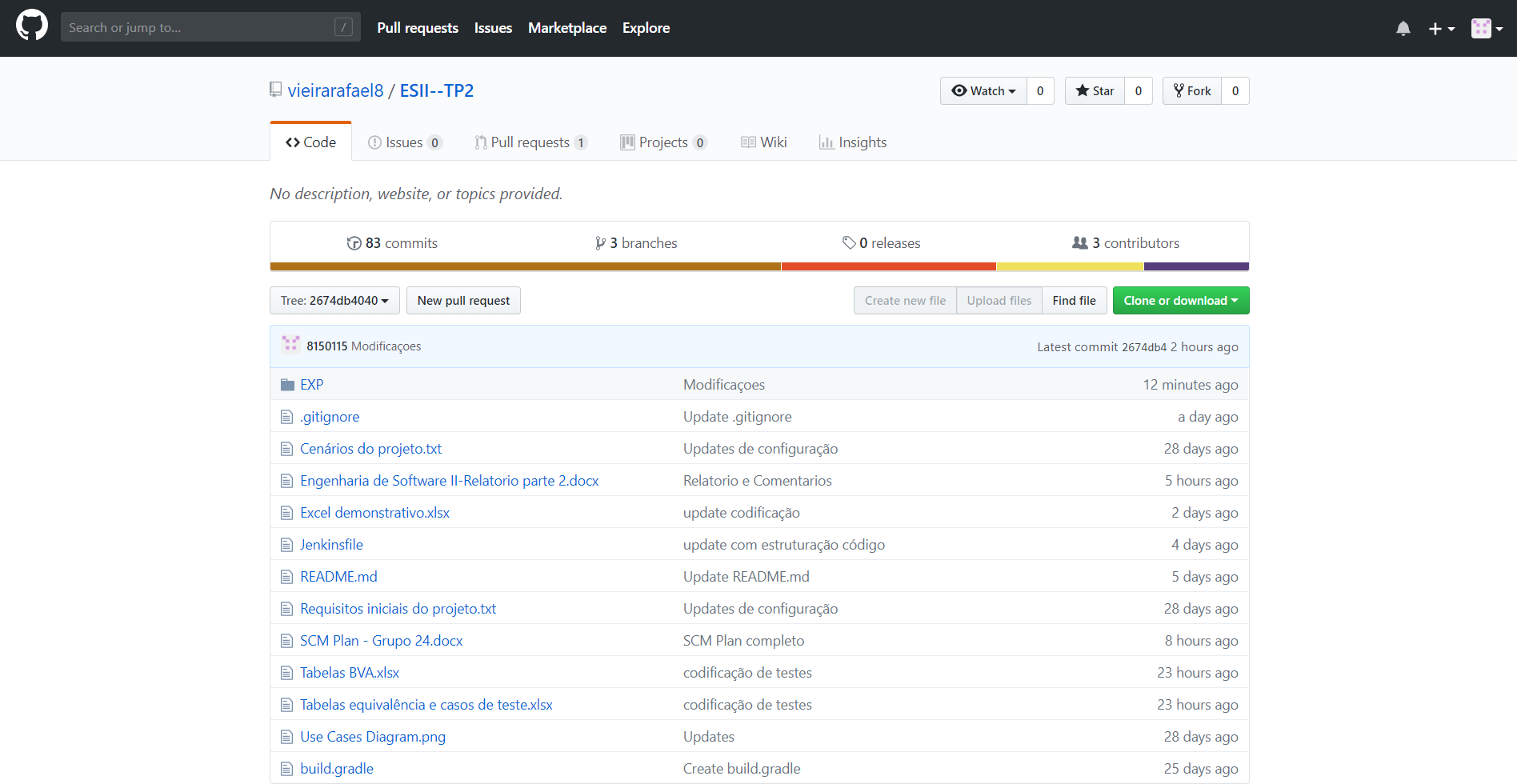
-O JUnit é um framework open source, que se assemelha ao sistema de testes java, com suporte para criação de testes automatizados na linguagem de programação Java.

Esse framework facilita a criação de código para uma automação de testes com apresentação dos resultados. Com este, pode ser verificado se cada módulo de uma série funciona da forma esperada, exibindo os erros e as provas podem ser usadas tanto para a execução de baterias de testes como para a extensão.

Com o JUnit, o programador tem uma possibilidade de usar esta ferramenta para criar um modelo padrão de testes, muitas vezes de forma automatizada.



-O GitHub é uma plataforma de hospedagem de código-fonte com controle de versão usando o Git. Ele permite que programadores, utilitários e qualquer usuário cadastrado na plataforma contribuam em projetos privados e / ou Open Source de qualquer lugar do mundo. O GitHub é amplamente utilizado por programadores para a divulgação de seus trabalhos ou para outros programas que contribuem com o projeto, além de enviar relatórios de forma facilitada através de recursos que relatam problemas ou mesclam repositórios remotos (questões, pull request).



# Project Management

Começamos com o brainstorm inicial do projeto, em que em grupo discutimos como iria ser realizado este projeto, o que cada um iria pesquisar para um estudo mais aprofundado sobre o tema e a resolução do mesmo. De seguida realizamos a defenição de requisitos, em que começamos com a Fase A que foi a fase de preparação que englobou a elaboração de pesquisas de métodos e criação de matrizes. De seguida foi a Fase B de cálculo em que pensamos em como utilizar as formulas nos metodos, por último Fase C em que elaboramos a apresentação dos resultados obtidos. De seguida definimos os Uses Cases a ser implementados no projeto. Em seguimento defenimos a elaboração de relatório de testes, e estudo das ferramentas a utilizar no projeto.

No sprint #2 elaboramos o SCM Plan, onde enunciamos o que iriamos elaborar e as ferramentas que usamos, os metodos que implementamos, as baselines, os CCB’s e as milestones.

No sprint #3 foi a realização deste relatório final.

# Project Development

No sprint #1 elaboramos as tabelas de equivalencia e de BVA’s do nosso projeto, e criamos o repositório Git e configuração do mesmo.

No sprint#2 fizemos a defenição da configuração do estado do projeto, dos controlos, finalização do SCM Plan e criação dos método de que retiram os caracteres especiais e o método que retiram os digitos.

No sprint#3 configuramos o jenkins e começamos a parte de codificação dos métodos e e elaboração dos testes.

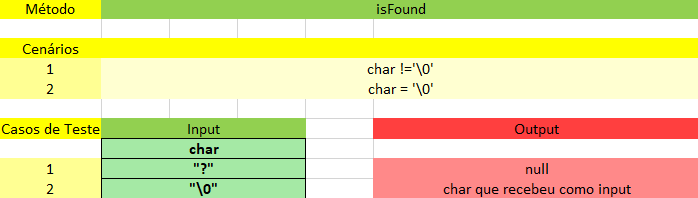
# Estudo e ánalise dos Métodos

# Metódo isFound

## Tabela ECP



## Tabela BVA



## 

## Casos de teste

## Caso de teste 1.1 ()

Se forem recebidos ficheiros, o programa deve ler os ficheiros e remover os caracteres presentes nos ficheiros.

O programa deve devolver os ficheiros sem os dígitos.

## Caso de teste 1.2 ()

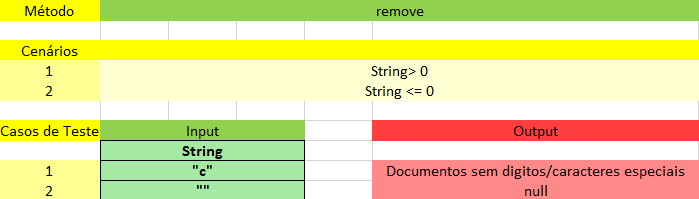
Se não for recebido nenhum ficheiro, o programa deve devolver -1.

# Metódo remove

## Tabela ECP



## Tabela BVA



## Casos de teste

## Caso de teste 2.1 ()

Se forem recebidos ficheiros, o programa deve ler os ficheiros e remover os dígitos presentes nos ficheiros.

O programa deve devolver os ficheiros sem os dígitos.

## Caso de teste 2.2 ()

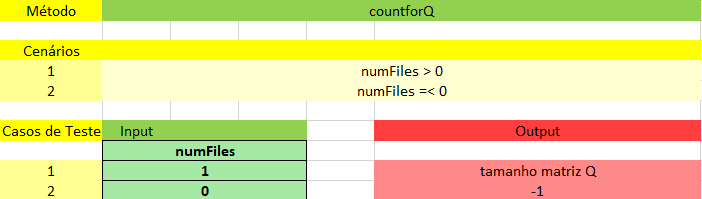
Se não for recebido nenhum ficheiro, o programa deve devolver -1.

# Metódo TratamentoPalavras

## Tabela ECP



## Tabela BVA



## 

## Caso de teste

## Caso de teste 3.1 ()

Se forem recebidos documentos, o programa deve ler os documentos e criar a matriz M que contem o número de ocorrências de cada palavra em cada documento.

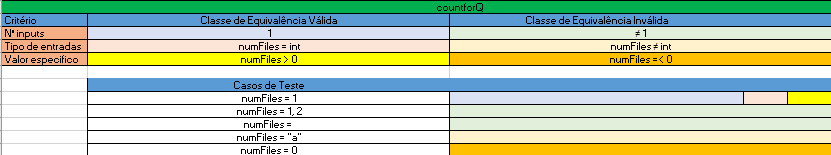
O programa deve devolver a matriz M.

## Caso de teste 3.2 ()

Se não for recebido nenhum documento, o programa deve devolver -1.

# Metódo countforQ

## Tabela ECP



## Tabela BVA

## 

## Caso de teste

## Caso de teste 4.1 ()

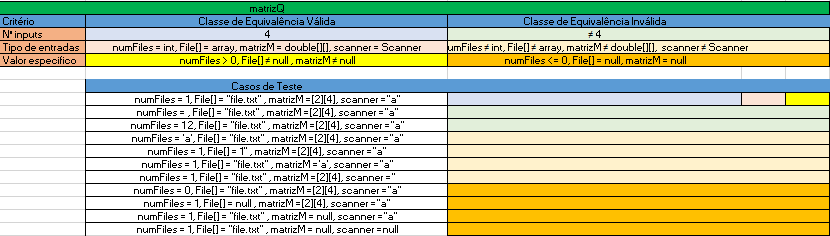
## Se o numero de ficheiros recebidos for um valor inteiro superior a 0, o programa deve devolver o tamanho da matrizQ para este ser utilizado na criação da mesma.

## Caso de teste 4.2 ()

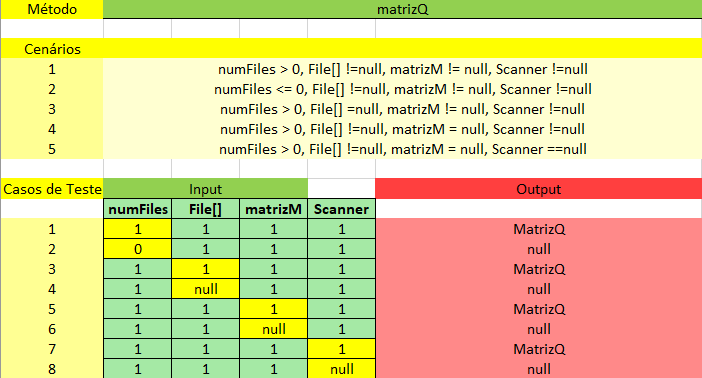
## Se o numero de ficheiros recebidos não for um valor inteiro ou for igual ou menor que 0, o programa deve devolver -1

## Metódo matrizQ

## Tabela ECP



## Tabela BVA



## Caso de teste

## Caso de teste 5.1 ()

Se o programa receber a matriz M, o numero de documentos e o numero de documentos em que a palavra p aparece, o programa deve, através da formula designada, criar a matriz arrayQ.

O programa deve devolver a matriz arrayQ.

## Caso de teste 5.2 ()

Se não for recebida a matriz M, o programa deve devolver -1.

## Case de teste 5.3 ()

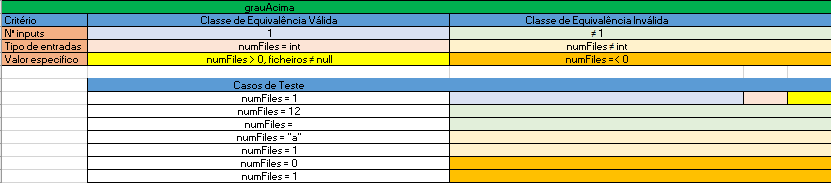
Se não for recebido o numero de documentos, o programa deve devolver -1.

## Caso de teste 5.4 ()

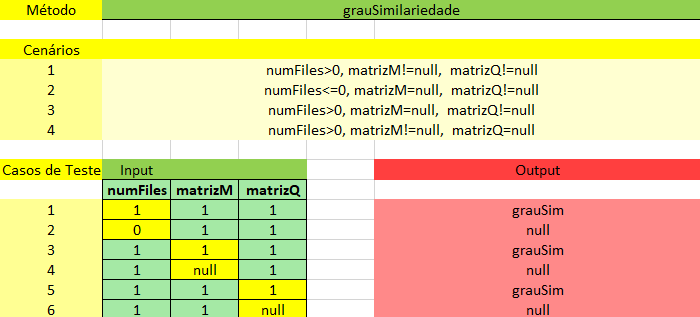
Se não for recebido o numero de documentos em que a palavra p aparece, o programa deve devolver -1.

# Metódo grauSimilariedade

## Tabela ECP



## Tabela BVA



## Caso de teste

## Caso de teste 6.1 ()

Se o programa receber a matriz M e a matriz array Q, o programa deve, através da formula designada, calcular o grauSimilaridade.

O programa deve devolver o grauSimilariedade.

## Caso de teste 6.2 ()

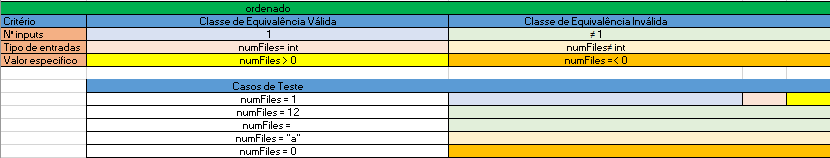
Se não for recebido o arrayQ, o programa deve devolver -1.

## Caso de teste 6.3 ()

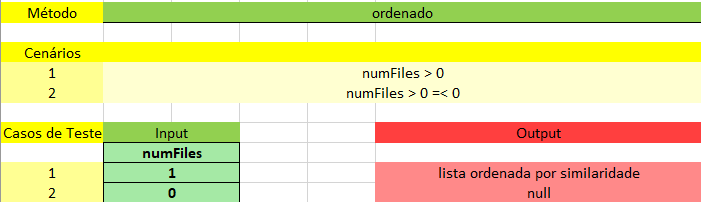
Se não for recebida a matriz M, o programa deve devolver -1.

# Metódo ordenado

## Tabela ECP



## Tabela BVA



## Caso de teste

## Caso de teste 7.1 ()

Se o programa receber a lista com os rankings de similaridade (ordenado), o programa deve ordenar a lista pelos respetivos rankings.

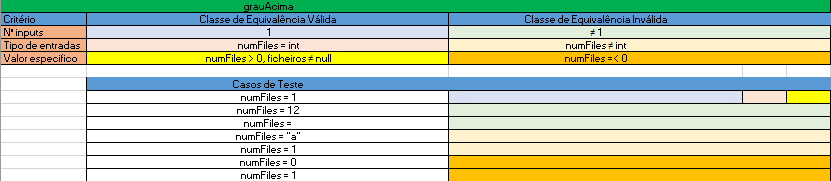
O programa deve devolver a lista ordenada por similaridade.

## Caso de teste 7.2 ()

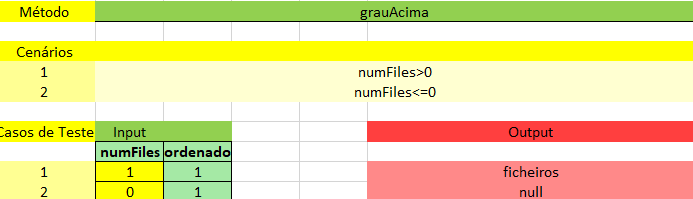
Se não for recebida a lista de ordenado, o programa deve devolver -1.

# Metódo grauAcima

## Tabela ECP



## Tabela BVA



## Caso de teste

## Caso de teste 8.1 ()

Se o programa receber a lista de ordenados e um valor para o grau que se quer definir, o programa cria uma lista com os ficheiros que têm um grau de similaridade acima do definido.

O programa deve devolver a lista com similaridade acima do grau.

## Caso de teste 8.2 ()

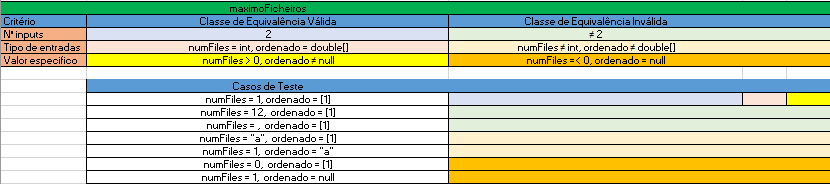
Se não for recebida a lista de ordenados, o programa deve devolver -1.

## Caso de teste 8.3 ()

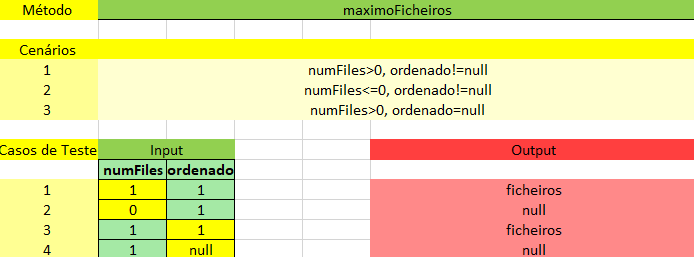
Se o valor do grau para criar a lista for inferior a zero, o programa deve devolver -1.

# Metódo maximoFicheiros

## Tabela ECP



## Tabela BVA



## Caso de teste

## Caso de teste 9.1 ()

Se o programa receber a lista de ordenados e um valor para o máximo de ficheiros que se quer ver, o programa cria uma lista com o número de ficheiros, ordenados pela similaridade, igual ao valor máximo definido.

O programa deve devolver a lista com máximo de ficheiros.

## Caso de teste 9.2 ()

Se não for recebida a lista de ordenados, o programa deve devolver -1.

## Caso de teste 9.3 ()

Se o valor máximo de ficheiros que se quer ver for igual ou menor que zero, o programa deve devolver -1.

# Conclusão

O segundo trabalho de ESII apresentou algumas dificuldades devido a problemas externos relacionados com as maquinas virtuais que acabaram por não poder ser utilizadas e sendo estas substituidas por uma outra maquina virtual partilhada pelos grupos ue se propuseram á realização deste trabalho prático.

Este projeto tambem enfrentou algumas dificuladades na implementação do codigo devido a erros nas formulas fornecidas no enunciado que foram mais tarde corrigidos.

Neste projeto obtivemos experiência na aréa de Engenharia de Software, com um forte foco nas metodologias ageis, sendo que a base para o desenvolvimento deste projeto foi o SCRUM (metodologia agile), e conseguimos adequirir competências práticas que nos permitiram evoluir e melhorar o nosso conhecimento sobre Software management e development.

# Anexos

# Github

* Software desenvolvido;
  + Necessária alteração no file Tratamento.java, na variavel global File,para o diretório do projeto do workspace atual;
* Product Backlog;
* JavaDoc;
* Tabela BVA;
* Tabela ECP;
* SCM Plan;
* Excel Demonstrativo Das Formulas;
* Use Case diagram;
* Apresentação em video;
* Coverage Report dos testes efectuados;
* Requisitos iniciais do projeto;
* Cenários do projeto;

**Anexos podem ser consultados:**

Repositório Git na plataforma Github através do seguinte Link:

[https://github.com/vieirarafael8/ESII--TP2.git](https://github.com/vieirarafael8/ESII--TP2.git?fbclid=IwAR2XxP3NlxIZ-aGAqGQHfA42tHXuXzJJvocNyDn5xbmbDglC7vOJwXe3cpQ)

# Máquina Virtual

* Integração com repositório Git;
* Pipelines;

**Anexos podem ser consultados:**

Máquina virtual acessivel através do seguinte IP(grupo 24):

172.20.10.17:8080