Licenciatura em Engenharia Informática

Graphical user interface

Description automatically generated with low confidence

Relatório Trabalho Prático 4

João Marques (1192221)

Paulo Couto (1200587)

Janeiro 2022

Table of Contents

[US 201 4](#_Toc89629083)

[Analise de complexidade 7](#_Toc89629084)

[US 202 8](#_Toc89629085)

[Analise de complexidade do método principal da *user story* 10](#_Toc89629086)

[Analise de complexidade do método *nearstPort* 11](#_Toc89629087)

[Analise de complexidade do método *findNearestNeighbour* 11](#_Toc89629088)

US 401

- As a Traffic manager I wish to know which ports are more critical (have greater centrality) in this freight network.

Nesta User Story, foi-nos pedido que como um Gestor de Tráfego, fosse possível ver quais são os portos mais críticos, ou seja, que têm maior centralidade nesta rede de frete.

Para tal, criámos o método getNCentralPorts, que recebe um Grafo com uma posição do navio e a sua distância, e com um inteiro referente ao número de portos que se pretende saber que têm maior centralidade.

Text

Description automatically generated

Figura 1 - Class Centrality ,method getNCentralPorts

## Analise de complexidade

A complexidade da US 201 é então:

**O(log n) + O(n) = O(n)**

# US 402

- As a Traffic manager I wish to know the shortest path between two locals (city and/or port).

Nesta user story era pedido que enquanto gestor de tráfego fosse possível encontrar o caminho mais curto entre dois locais. Podendo esses locais ser uma cidade e/ou um porto.

Text

Description automatically generated

Figura 2- Class RolesUI, method shortPathMenu

Text

Description automatically generated

Figura 3 - continuation of method shortPathMenu

Text

Description automatically generated

Figura 4 - continuation of method shortPathMenu

Text

Description automatically generated

Figura 5 - Class Algorithms, method shortestPath

*Text

Description automatically generated*

Figura 6 - Class PositionMatrixGraph, methods getLandMap and getSeaMap

## Analise de complexidade do método principal da *user story*

A complexidade deste método é de O(n) pois temos um *if* com seguido de um *else*, sendo que o *if* que se encontra no seu interior tem uma complexidade de O(1) a complexidade é como antes referido O(n).

# US 403

- As a Traffic manager I wish to know the most efficient circuit that starts from a source location and visits the greatest number of other locations once, returning to the starting location and with the shortest total distance.

Nesta user story era pedido que enquanto gestor de tráfego fosse possível encontrar o circuito mais eficiente que começa numa localização, visita o maior número de outras localizações uma única vez, retornando à mesma localização inicial com a menor distância total possível.

Para tal criámos o método mostEfficientCircuit, que cria uma Lista de LinkedLists de Posições recorrendo à utilização do algoritmo allCycles. O allCycles utiliza o algoritmo allPaths, que retorna todos os caminhos possíveis, sendo estes caminhos uma sequência alternante de vértices adjacentes e os seus ramos e que não contém nenhum ramo repetido. Em seguida, allCycles utiliza estes caminhos todos e verifica neles todos os circuitos que passam no maior número de vértices. Um circuito é um caminho fechado que não contém qualquer ramo repetido.

# Text Description automatically generated

Figura 7 - Class Circuit, methods mostEfficientCircuit and pathDistance

Text

Description automatically generated

Figura 8 - Class Circuit, method print

Text

Description automatically generated

Figura 9 - Class Algorithms, methods allCycles and allPaths