Daniel Montezuma A00382231

Diego Fernando Mueses Zuñiga A00382021

Entrega preliminar:

Fase 1:

In any developed or developing city, the main affectation can be seen in the mobility which is influenced by very bad road logistics, so much so that it can be something very overwhelming and stressful for those cleaning entities that need to do a highly efficient work in terms of time. Within this problem it can be evidenced that organizations that suffer from bad time logistics end up with great losses of money and reputation, damaging more and more the services of the company and in some cases causing its temporary or definitive closure.

However, when it comes to designing efficient routes in a city without repeating streets or looking for the shortest route from one point to another, for example for the garbage truck (to save time, money and pollution), the implementation of graphs with their methods, is of great help to solve these errors of road logistics that a cleaning company has and thus allow you to reduce costs with the effective optimization provided by the application.

1. To help the cleaning service route in the city to be much more efficient in the city, the system implements the Floy and Warshall algorithms to display the route that allows it to pass between all the streets in the city in the shortest possible time.
2. To make sure that the route of the cleaning service in the city is effective, the system implements the BFS and DFS algorithms to make sure that the service goes through all the streets in the city.

REQUIREMENTS

| Client | Cleaning Company |
| --- | --- |
| User | Company employees |
| Functional Requirements | R1: The system will allow storing the streets entered by the user.  R2: The system will allow adding connections between streets, which function as paths for the users.  R3: The system will allow verifying that all streets are connected, thus making the route much more effective.  R4: The system will verify if the entire city can be traversed by passing only once through each registered street.  R5: The system will be able to delete an entire route, allowing the user to try different routes without closing the application.  R6: The system will allow finding the shortest path from one selected street to another.  R7: The system will verify if it is possible to go through the whole city without repeating streets. |
| Context | A cleaning company that wants to streamline its cleaning services using methods such as knowing the best route, the shortest route, the most efficient route or if you can go through the whole city without passing twice through a street, all this with the data that we will enter and store in our system. |
| Non-Functional requirements | RNF1: Efficiency, the program will be able to operate properly according to the data entered by the user.  RNF2: Usability, the program will use help and error messages that are informative to the user. |

Fase 2:

Elicitation of requirements:

**R1**. The system will allow storing the streets entered by the user.

**R2**. The system will allow adding connections between streets, which function as paths for the users.

**R3**. The system will verify that all streets are connected, thus making the route much more effective.

**R4**. The system will verify if the entire city can be traversed by passing only once through each registered street.

**R5**. The system will be able to delete an entire route, allowing the user to try different routes without closing the application.

**R6**. The system will allow finding the shortest path from one selected street to another.

**R7**. The system will verify if it is possible to go through the whole city without repeating streets.

**Aids found for the solution of the problem**

For requirement 6 the Floyd and Warshall algorithm is of great help since among its characteristics are:

* It obtains the best route between all pair of nodes.
* It works with the matrix 'D' initialized with the direct distances between every pair of nodes.
* The iteration occurs over intermediate nodes, i.e., for every element of the matrix is tested if the best to go from 'i' to 'j' through an intermediate node chosen or as it was previously, and this is tested with all nodes in the network.
* Once all nodes in the network have been tested as intermediate nodes, the resulting matrix gives the best distance between every pair of nodes.
* Matrix 'Sn' that gives the intermediate node to get from node 'i' to node 'j' of the network.

For requirement 7 what we can do is the implementation of BFS and check if the network is strongly connected.

Bibliography:

https://sites.google.com/site/aplicaciongrafos/

Fase 3:

Brainstorming is a method of generating ideas and sharing knowledge to solve a particular commercial or technical problem, in addition to actively stimulating the generation of ideas even if they are not possible. This contributes as a team to find the solution to the problem.

1. Develop an app with an interface in javaFX.
2. Design a system that allows adding streets and routes.
3. Design a system that allows adding a connection between two streets.
4. Develop a system that allows you to see the shortest path in a route.
5. Design a system that allows you to delete a route
6. Design a system that allows you to verify if you can get to all the streets from any street.
7. Design a system that allows editing streets and some conditions. For example, if the street is full, in reconstruction processes. This helps to avoid that street and make the route efficient.
8. Design a system with an interface in javaFX that allows you to see the shortest route in real time.
9. Design a system with an interface in javaFX that allows you to see a map of all the streets that the user has added.
10. Design a system that allows to see if the route of a city can be done without repeating streets.
11. Design a system that supports multiple routes.
12. Design a system that allows you to go through all the streets of a city in the shortest possible time (or weight in the case of graphs).
13. Implement a single type of graph that allows the use of the Floyd and Warshall methods and the BFS and DFS as appropriate.
14. work with 2 different types of graphs.

Fase 4:

Discarded ideas:

8.Design a system with an interface in javaFX that allows you to see the shortest route in real time.

9.Design a system with an interface in javaFX that allows you to see a map of all the streets that the user has added.

These are very interesting ideas, but we would have to work with external java libraries to graph the graph and this will take too much time.

14.work with 2 different types of graphs.

This is not a good idea, we would have to carry out the operation twice because of that. On the other hand, we could just use a pondered graph, and this will allow us to make all the processes in only one graph.

Great ideas:

1. Design a system that allows editing certain streets and adding conditions to it, for example if it is full or in reconstruction process, etc. To avoid traffic on that street and avoid

We will make this with the pondered graph, the conditions will be the weight of the edges, the higher the weight, the higher the road “condition”.

1. To allow them to work with different routes.

We will make this with the option of delete route, this will allow the user to work with different types of routes.

1. Developing an application with a javaFX interface is a very good idea but we will only take it with the following features:

add streets

add routes

delete streets

check route.

Know the shortest route.

Find out if you can tour the city without repeating streets

1. Implement a single type of graph that allows the use of the Floyd and Warshall methods and the BFS and DFS as appropriate.

This will make all the process easier as we explained in the discarded ideas

Implementation ideas with the discarded ideas and the feasible ones:

**Idea 1:**

A program that will work with only one type of graph (pondered graph). This program will allow the user to add a street, add a connection, verify a route, verify connection, delete route,delete street and find the shortest path. All of those options will be displayed with an interface. Once the user has selected the delete route option, the user can restart the process with a different route.

**Idea 2:**

A program that will work with 2 different types of graphs to work with the different algorithms implemented in the app. This program will allow the user to work only with 1 route. If the user wants to change the route, he must close the app and restart with the process. this app will show all the options as add, delete, verify..etc. the program will display all of these options in the console.

**Idea 3:**

A program that will work with 2 different types of graphs to work with different algorithms implemented in the app. This program will display the options in the console as Add, Verify, Shortest path and delete connection, edit street. This program won't allow the user to delete a street only edit. If the user wants to delete the street or change all the route, he must close the app.

Fase 5:

The selection criteria are established:

**Criterion A. Precision of the solution.**

[2] exact

[1] Approximate

**Criterion B. Efficiency.**

[3] exact

[2] Approximate

[1] null

**Criterion C. Completeness.**

[3] exact

[2] Approximate

[1] null

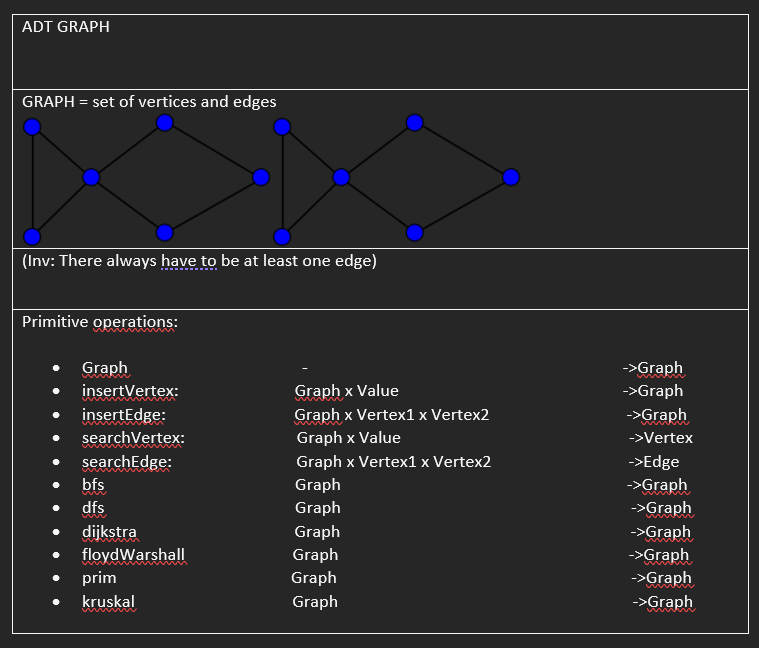
**Criterion D. Ease of algorithmic implementation:**

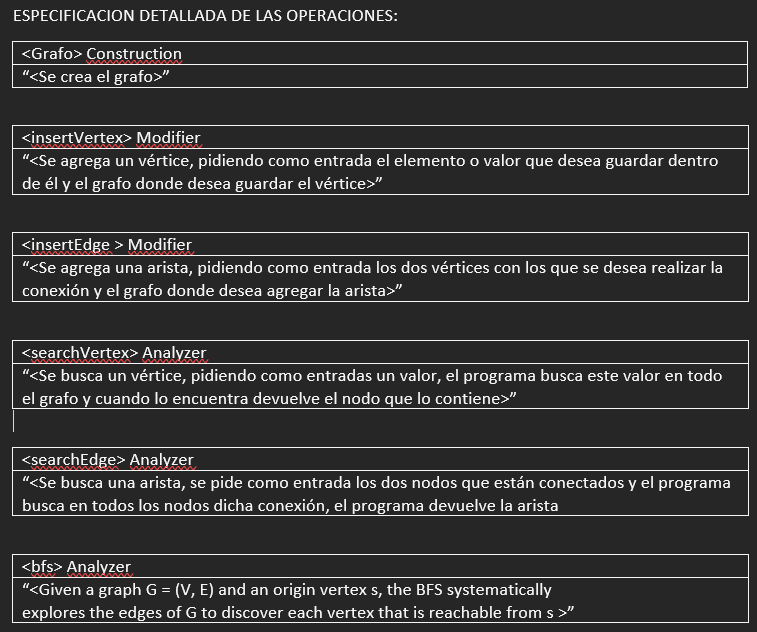
[2] Compatible with the basic arithmetic operations of a modern computer equipment

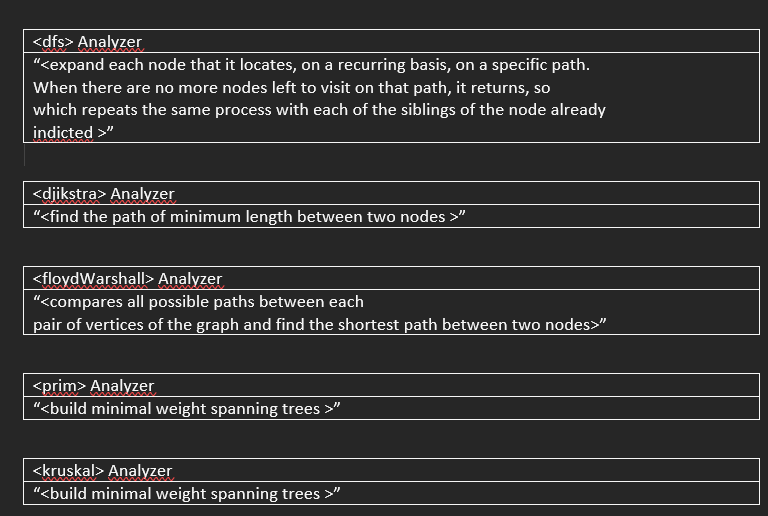
[1] Not fully compatible with the basic arithmetic operations of a modern computer

|  | Criterion A | Criterion B | Criterion C | Criterion D | Total |
| --- | --- | --- | --- | --- | --- |
| Idea 1 | [2] | [3] | [2] | [2] | 9 |
| Idea 2 | [1] | [2] | [1] | [2] | 6 |
| Idea 3 | [1] | [1] | [1] | [2] | 5 |

1. Especificación del TAD:







1. Problem to solve:

A)

An application will be developed that can help the workers of the cleaning service of the city of Cali, to verify routes which allow them to reach all the streets, from any street. Also, the system will be able to find the shortest path from one street to another, to make the cleaning system more effective. The application will have a menu that will allow workers to select different options such as:

\*Add street

\*Add a connection

\*Find shortest path

\*Verify connection

\*Verify route

\*Delete route



The application works with pondered graphs. Users will be able to add a street only by selecting the add street option and registering the street name. To make a connection between streets, users will select the option to connect a street and then type the name of the two streets in which they wish to make the connection (Edge). To find the shortest route that can be taken to get from one street to another, the application will work with the Floy and Warshall algorithm, users will only have to click on the option to find the shortest distance presented by the program. If users want to see if there is a path between one street and another, they can select the check connection option which works basically like an edge. Users will be able to verify that their route is capable of reaching a point from any point, to stop this they only have to select the verify route option, which works with the BFS algorithm. If users want to add or change routes, they can select the delete route option, which removes all streets and connections between them. If users wish to eliminate a connection or path between two streets, they can select the option to eliminate connection and enter the name of the two streets in which they wish to eliminate the path or connection.

B)

Functional Requirements:

1. The system will allow the streets entered by the user to be stored.
2. The system will allow the addition of connections between streets, which function as paths for users.
3. The system will make it possible to verify that all the streets are connected, thus making the route much more effective.
4. The system will make it possible to verify if the entire city can be covered by passing through each registered street only once.
5. The system will be able to delete an entire route, allowing the user to try different routes without closing the application.
6. The system will allow you to find the shortest path from a selected street to another.