Names: Brayan Ortiz Bastidas, Gilmar Andrés Amézquita Romero, Jhan Carlos Carvajal Bastidas

## *PHASE 1: PROBLEM IDENTIFICATION*

You want to develop a software system that allows the users to visualize the shortest route between two cities, either to define delivery routes or trips between them. To make it practical, the first version of this software product will contain only data about municipalities that belong to the Valle del Cauca department.

### *Functional requirements*

**Req1.** Read the database. The program must be able to read a file with information which contains the name of cities, the ways between them and also the length of those ways.

**Req 2.** Allows the user to select two cities to calculate the shortest way between them.
**Req 2.1.** Allows the user to select the initial city, the destiny city and the cities that must appear in the shortest calculated route.

**Req 3.** Show a Valle del Cauca's map with the municipalities and the ways that exist between them. The program must show the map when it starts.
**Req 3.1.** Show the map but highlighting the shortest route.

## *PHASE 2: COMPILATION OF NECESSARY INFORMATION*

*Graph:* Graph is a set of points and ways that connect them.

The Valle del Cauca department has 42 municipalities, which means that in our program will appear 42 cities enable to select and calculate the shortest route. The 42 cities are the following:

| | | |
|---|---|---|
| Andalucía | Tuluá | Cartago |
| Bugalagrande | Yotoco | El Dovio |
| El Cerrito | El Águila | Obando |
| Ginebra | Alcalá | Roldanillo |
| Guacarí | Ansermanuevo | Toro |
| Buga | Argelia | Versalles |
| Riofrío | El Cairo | Zarzal |
| Trujillo | La Unión | Buenaventura |
| Calima – El Darién | La Victoria | Caicedonia |
| Restrepo | Ulloa | Sevilla |
| San Pedro | Bolívar | Cali |

Names: Brayan Ortiz Bastidas, Gilmar Andrés Amézquita Romero, Jhan Carlos Carvajal Bastidas

| | | |
|---|---|---|
| Candelaria | Florida | Pradera |
| Palmira | Jamundí | Vijes |
| Dagua | La Cumbre | Yumbo |

## *PHASE 3: SEARCH FOR CREATIVE SOLUTIONS*

Keeping in mind the best way to solve this problem we were proposed some alternatives:

**Alternatives to find the shortest route:**

*Alternative A:* Search for the shortest route using the Dijkstra algorithm.

*Alternative B:* Search for the shortest route using the Floyd-Warshall algorithm.

**Alternatives to show the shortest route:**

*Alternative 1*: Displays a 3D relief Valle del Cauca map and over it the trajectory between the two cities that were been selected by the user.

Alternative 2: Display a 2D map that shows the shortest route between the selected points.

*Alternative 3:* Display a list through console where the user will can see the order of the cities that compose the shortest route.

## *PHASE 4: TRANSFORM THE IDEAS FORMULATION TO THE PRELIMINAR DESIGNS*

First of all, we discard the alternative 1 since its complexity level our current knowledge. On the other hand, analyzing in detail the two other alternatives we have the following:

Alternative 2:

- It is simple to implement, and it allows to accomplish the proposed objectives.
- It shows in minimalism way the shortest route between the two cities selected by the user.
- Its simplicity may do it less realistic and less useful.

Alternative 3:

- It does not give a visual representation where the users can see the routes in a comfortable way.
- It does not allow to accomplish one of the requirements which says that the program must show to the user the routes between the cities selected.

- It is easy to implement.

## *PHASE 5: EVALUATION AND SELECTION OF THE BEST SOLUTION*

We evaluated the proposed alternatives to select the best option:

Criterion 1: Compliance with requirements
[1]: It does not comply with the proposed requirements
[2]: It comply the majority of the proposed requirements
[3]: It comply every proposed requirement

Criterion 2: Route visualization
[1]: Shows calculated routes without any visual representation
[2]: Shows in a general way the calculated routes
[3]: Shows the calculated routes in great detail

Criterion 3: Ease of implementation
[1]: Exceeds the current knowledge of the members
[2]: It is easy to implement, but does not have a visual representation
[3]: It is simple to implement and has visual representation

| | Criterion 1 | Criterion 2 | Criterion 3 | Total |
|---|---|---|---|---|
| Alternative 2 | 3 | 2 | 3 | 8 |
| Alternative 3 | 2 | 1 | 2 | 5 |

According to the proposed criteria, we choose as the best option the alternative 2 since it got the best evaluation.

## *PHASE 6: PREPARATION OF REPORTS AND SPECIFICATIONS*

Problem: You want to develop a software system that allows the users to visualize the shortest route between two cities, either to define delivery routes or trips between them. To make it practical, the first version of this software product will contain only data about municipalities that belong to the Valle del Cauca department.

Names: Brayan Ortiz Bastidas, Gilmar Andrés Amézquita Romero, Jhan Carlos Carvajal Bastidas

Expected input: A file with the name of the cities, ways between the cities and length of those ways.

Expected output: A graphical representation of the shortest route between two cities.

## *PHASE 7: DESIGN IMPLEMENTATION*

| | | |
|---|---|---|
| **Name:** GraphND | | |
| **Definition:**<br><br>E is the adge.<br>V is the vertex. | | |
| **Input:** {key: the value used to manage and search the vertex; Value: the value that must be stored} | | |
| **Invariable:** | | |
| **Operations** | | |
| **Name** | **Input** | **Output** |
| GraphND<br>**constructor** | | Create distance Matrix empty |
| addVertex<br>**modifier** | V | Add the vertex to the list of vertexes |

| | | |
|---|---|---|
| addEdge<br>**modifier** | {V, V, Double} | Add the edge with his weight in the list of edges |
| getRouteByDK<br>**analyzer** | {V, V} | Return the shortest route |
| getRouteByFW<br>**analyzer** | {V, V} | Return the shortest route |

***Operations***

| **Operation:**<br>GraphND | | | |
|---|---|---|---|
| **Operation type:** constructor | | | |
| **Input** | **Output** | **Precondition** | **Postcondition** |
| | Distance matrix | true | Distant matrix is empty |
| Create distance matrix empty | | | |

| **Operation:**<br>addVertex |
|---|

| **Operation type:** modifier | | | |
|---|---|---|---|
| **Input** | **Output** | **Precondition** | **Postcondition** |
| | List of vertexes with de new vertex | Listo of vertices initialiced | List =/= initial size |
| Add the vertex to the list of vertexes | | | |

| **Operation:**<br>addEdge | | | |
|---|---|---|---|
| **Operation type:** modifier | | | |
| **Input** | **Output** | **Precondition** | **Postcondition** |
| | List of edges with de new edge | List of edges initialiced | List =/= initial size |
| Add the edges to the list of edges | | | |

| **Operation:**<br>getRouteByDK | | | |
|---|---|---|---|
| **Operation type:** analyzer | | | |

Names: Brayan Ortiz Bastidas, Gilmar Andrés Amézquita Romero, Jhan Carlos Carvajal Bastidas

| **Input** | **Output** | **Precondition** | **Postcondition** |
|---|---|---|---|
| | Min routes evaluate with Dijkstra method | True | The route retuned is the shortest route |
| Return the shortest route | | | |

| **Operation:**<br>getRouteByDK | | | |
|---|---|---|---|
| **Operation type:** analyzer | | | |
| **Input** | **Output** | **Precondition** | **Postcondition** |
| | Min routes evaluate with Floyd warshall method | True | The route retuned is the shortest route |
| Return the shortest route | | | |