Gabriel Escobar | Vanessa Sánchez Morales | Luis Manuel Rojas
Computación y Estructuras Discretas I
Integrative Task II

REQUIREMENT ANALYSIS
MARLON-MANIA

REQUIREMENT ANALYSIS TABLE	
Client	Dr. Marlon Gómez Victoria
User	Players
Functional Requirements	• RF1: Start a new game. • RF2: View Top scores. • RF3: Place pipeline • RF4: Verify sewer system
Problem Context	*Marlon Mania game consists of a sewer system simulation. In this game, the player can locate three different types of "pipes" within an 8x8 board, with the objective of connecting the "water source" to the "draining pipe" in the most efficient way. The users can also view a best score ranking.*
Non-Functional Requirements	• The program must two different graphs implementations. • The program must use at least two different graph algorithms. • The program must be developed with a user interface.

<table>
<tr><td>Identifier and Name</td><td colspan="3">RF1: Start a new game</td></tr>
<tr><td>Summary</td><td colspan="3">To start a new game the player must give his nickname and choose the difficulty level.

1.Easy (User must connect source and drain) 2. Difficult (User must connect source and drain ensuring water moves from one point to another using the least amount of “feet”).

Then, the system will display an 8x8 board with an “F” and a “D”, each one representing the water source and the draining pipe respectively. Besides an option menu to play the game will be shown.</td></tr>
<tr><td rowspan="4">Input</td><td>Input name</td><td>Data type</td><td>Valid condition</td></tr>
<tr><td>User Nickname</td><td>String</td><td>Can´t be empty</td></tr>
<tr><td>Game Difficulty</td><td>Int</td><td>1.Easy
2.Hard</td></tr>
<tr><td>Graph</td><td>int</td><td>1. List
2. Matrix</td></tr>
<tr><td>Result or Postcondition</td><td colspan="3">After the system receives the data entered by the user, it will show the game options (place pipe, verify and exit) and the 8x8 board game, randomly locating an “F” and a “D”, each one representing the water source and the draining pipe respectively.</td></tr>
<tr><td rowspan="3">Output</td><td>Output name</td><td>Data type</td><td>Format</td></tr>
<tr><td>gameBoard</td><td>Graph</td><td>X X X X X X X X
X X X X X X F X
X X X X X X X X
X X X X X X X X
X X X X X X X X
X X X X X X X X
X D X X X X X X
X X X X X X X X</td></tr>
<tr><td>gamePlayMenu</td><td>String</td><td>1. x coordinate
2. y coordinate
3. Select pipe</td></tr>
</table>

			4. Add pipe *5. Simulate*

Identifier and Name	***RF2: View scores***		
Summary	*The system must show, in descending order, the final scores of the players that have played and finished a game.*		
Input	**Input name**	**Data type**	**Valid condition**
	N/A	N/A	N/A
Result or Postcondition	After each successful game, the system calculates the user's score. The system will calculate the player´s final score using the following formula: Score= (1000)-(50)*(number of pipes used or modified)		
Output	**Output name**	**Data type**	**Format**
	Final Score List		*1)PlayerNickname = Score*

<table>
<tr><td>Identifier and Name</td><td colspan="3">RF3: Place Pipes</td></tr>
<tr><td>Summary</td><td colspan="3">After entering the option to Start a new game menu (see RF1), The system must allow the user to locate a “pipe” in a specific position of the 8x8 board, by asking for the coordinates in which the new pipe will be located and the type of pipe.</td></tr>
<tr><td rowspan="4">Input</td><td>Input name</td><td>Data type</td><td>Valid condition</td></tr>
<tr><td>xCoordinate</td><td>int</td><td>Must be an Integer [0-7]</td></tr>
<tr><td>yCoordinate</td><td>int</td><td>Must be an Integer [0-7]</td></tr>
<tr><td>pipeType</td><td>int</td><td>1.Horizontal (=)
2.Vertical (||)
3.Circular(o)</td></tr>
<tr><td>Result or Postcondition</td><td colspan="3">The system searches for the coordinate that the player chose. If said coordinate is located within the possible range of the board and it isn´t occupied by an “F” or “D”, the type of pipe that the player chose will be displayed in the [x,y] coordinate of the board. Else, the board will appear with no changes made.</td></tr>
<tr><td rowspan="2">Output</td><td>Output name</td><td>Data type</td><td>Format</td></tr>
<tr><td>gameBoard</td><td>graph</td><td>8x8 board of “x” characters, contains an F (water source), a D (draining pipe) and the pipes (“=”,”||”,”o”)</td></tr>
</table>

<table>
<tr><th>Identifier and Name</th><td colspan="3">RF4: Verify sewer system</td></tr>
<tr><th>Summary</th><td colspan="3">The system must verify that the pipes' solution provided by the user is valid, meaning, the water source ("F") is connected to the draining pipe ("D") with a correct usage of the pipes ("=","||","o"). Then, the user will see if their option is correct or not, and depending on that, the game will close.</td></tr>
<tr><th rowspan="2">Input</th><th>Input name</th><th>Data type</th><th>Valid condition</th></tr>
<tr><td>N/A</td><td>N/A</td><td>N/A</td></tr>
<tr><th>Result or Postcondition</th><td colspan="3">The system checks that the pipes are located correctly according to their type and direction. The "F" and "D" must be connected, allowing the water flow, with the "=" pipes going one next to the other, and the "| |" pipes going one under the other. Besides, an "o" cannot be next to another "o" or to the "F" and "D", and it can only be used to do a 90° spin with the pipes.

If the solution is correct, the game is closed, the score calculated and saved; else, the game continues, and the menu and board will be displayed again.</td></tr>
<tr><th rowspan="2">Output</th><th>Output name</th><th>Data type</th><th>Format</th></tr>
<tr><td>message</td><td>String</td><td>Whether the user´s option was correct or not.</td></tr>
</table>