**Functional Requirements**

**FR1: Allow the creation of a game board using recursion. The user is prompted to enter the desired dashboard size, getting dimensions of n rows and m columns. In addition, a number of snakes and a number of e stairs will be requested. Each player will use a different symbol for their respective identification (\*! Or X % $ + &).**

**FR2: Allow you to start a game. The user will be able to start the game, players will start in box 1 and move in turns, each throws the dice and then their snake will move to the corresponding box. Stairs will be randomly located considering that no ladder can start in box 1 and that no stair or snake start, or end box must match another start or end of ladder or snake.**

**FR3: Allow the dashboard to be displayed. When starting the game, the user will be able to see the game board represented by a grid formed by square brackets with the dimensions that entered above, each box will be listed and with the location of the stairs and snakes. The stairs will be shown in bold to be able to differentiate them from the numbers of the squares. The board will show player positions and stair locations, but the numbers of the squares will no longer be displayed.**

**FR4: Allow you to choose different options while playing. During the game, the user will be able to play manually by simply entering a line break so that the player who has the turn can move. If instead of entering the line break enter the word "num" and then the line break occurs, the program will display the same grid that it displays at the beginning, with the numbered boxes, snakes, and stairs. A line break will be expected to continue. If you enter the word "simul" and you jump in line, you will start in simulation mode, you will go showing what each player in their turn plays, with the corresponding board of each new position, waiting 2 seconds between each play, but without waiting any line exit. If on the other hand you type the word "menu" and jump the line, the game is cut unfinished and the program returns to the main menu, showing your options.**

**FR5: Allow to show the scores of each player. If a player wins the game and returns to the menu, the score of each of the participants is calculated, which will be equal to the number of moves multiplied by the total number of squares on the board. These scores will be stored in a binary search tree sorted inversely by score. Option 2 will display a list of player names or nicknames, with their scores and symbols, resulting from having traveled the binary tree in inorder.**