Algorithms and programming

Snakes and Ladders

Members

Luis Murcia

Diana Olano

Icesi University

Santiago de Cali, April 4, 2021

Luis Murcia and Diana Olano Icesi University

Functional requirements: Integrative Task 2

The program must be able to:

RF1*:* ***Start*** with a simple menu with 3 options. The first option is to play, the second option is to see the leaderboard and the third option is to exit the program. When the first option is chosen, the program will wait for 5 positive integers separated by space to be typed on the same line, indicating n, m, s, e and p respectively.

RF2: ***Create*** a game with a grid size n x m, with s snakes and e ladders located randomly combining any of the boxes of the board. No ladder starts at square 1, no snake starts at square nxm, and no start or end square of ladder or snake must coincide with another start or end of ladder or snake. The program will not allow you to create more than a quarter of the stairs of the total number of squares in the grid, nor more than a fifth of snakes.

RF3: ***Show*** a grid made up of square brackets, with the squares correctly numbered and with the location of the ladders and snakes after the user enters the game parameters mentioned above. Once this display of the grid is displayed, the program waits for a line break to start and display the first board. The game boards will show the position of the players in the squares, the program will not show the numbers of the squares, although the ladders and snakes will.

RF4: ***Assign*** randomly user symbols that will be used throughout the game or that can be chosen by each user.

RF5: ***Show*** the first board after the grid has been displayed and the user has entered a line break. The game boards will show the position of the players in the squares, but they should no longer show the numbers of the squares, although they do show the ladders and snakes.

RF6: ***Play*** if and only if a line break is entered, for the player whose turn corresponds to play. After this a random number will be generated between 1 and 6 and the program will advance the corresponding boxes. Then a message of the player symbol and the number of squares shifted will be displayed.

RF7: ***Return or advance*** a player respectively by landing on a square that performs these actions.

RF8: ***Show*** the same grid that shows at the beginning, with the numbered boxes, the snakes and the ladders if instead of taking a line break, first write the word **num**, and then a line break, after showing all the board, another line break will be waited to continue with the game.

RF9: ***Get in*** simulation mode, if you write the word **simul** and then jump the line, the program will start in simulation mode, which consists of showing what each player in turn plays, with the corresponding board of each new position, waiting 2 seconds between each play, but without expecting any salt t or line.

RF10: ***Cuts*** the game suddenly, if instead of simply entering a line break to the program, the word **menu** u is written and then a line break is made, the game is cut without ending and the program returns to the menu main, showing yourself your options.

RF11: ***Finish*** the game after that a player reaches the last box, in that case the program will display a message that it will say: " The Z player has won the game with Y movements ". Following this message, it will ask to the name or nickname of the winning player and then the main menu of the program shown.

RF12: ***Calculate*** the score of the winning player. This score is equal to the number of moves multiplied by the total number of squares on the board. This score will be stored in a binary search tree ordered inversely by score. Option 2 will show a list of the names or nicknames of the players, their symbols and their respective scores, the result of going through the binary search tree in order.

RF13: ***Report*** the user as many times as he needs to enter a line break.