UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Object Oriented Programming

Final Project: Nonogram

Team 1

Neftali Rebollo Mercado 167743

Álvaro Fernando Oros Ramírez 171621

Aldo Cordova Gonzalez 167343

Guillermo Francisco Ramírez Pérez 158434

May 16th 2021

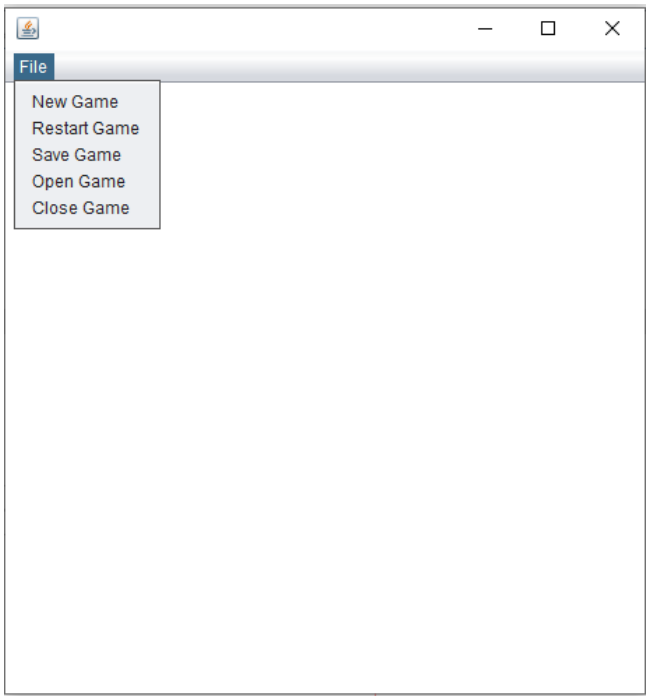
Spring 2021

Objective

The general objective of this project is to design, think, develop and apply all the techniques acquired through this course, to create with our knowledge and through self-investigation a famous computer game “Nonogram”. This project has as well other specific objectives, which were a vital part for the project to work properly. These specific objectives/tasks were the following: the user can choose to play between at least 2 different pre-charged games, the user can choose to save the current game and on the other hand to open a previous stores/saved game. In addition, for the interaction to be as if the user was playing, there are a fixed number of lives so that the user tries its best to completely solve the game.

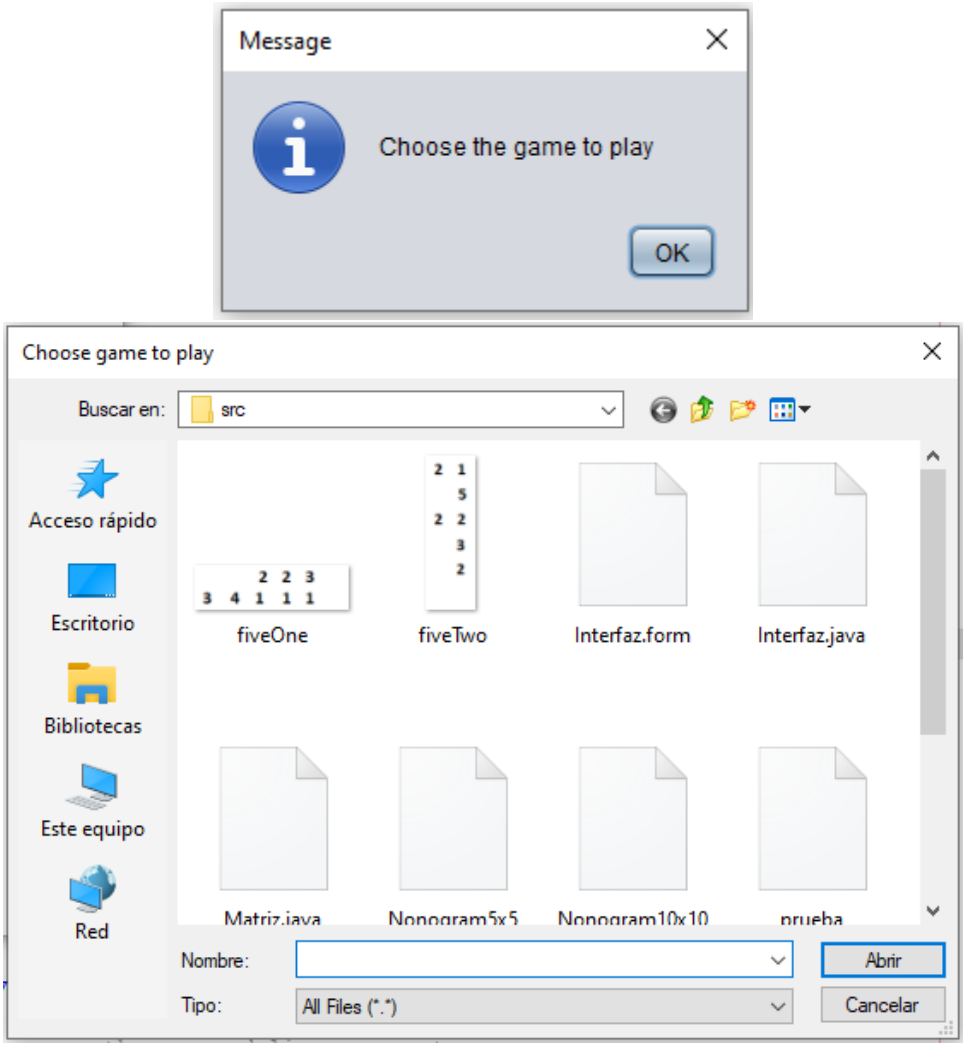
Program Explanation

Image 1.



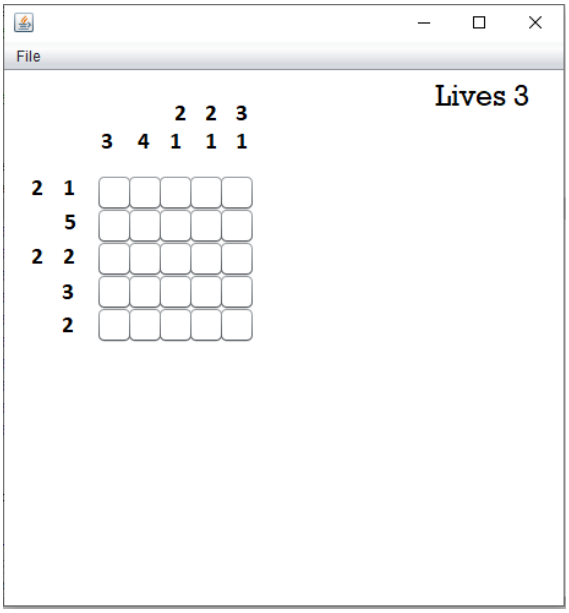
Once we run the program, the first window that the user will view is totally in blank, so the user needs to first click on the File menu in order to view the command options that he has.For the user to start playing, he has to click on New Game option.

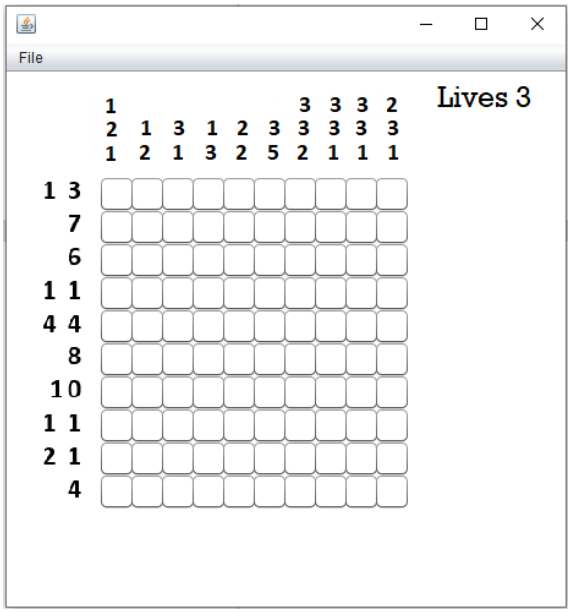
Image 2.



As the user is initiating the game, he'll get a message indicating that he needs to choose the game to play, once he clicks on OK button, another window will open with all the files of the game's folder inside the projects folder; in our program we have 2 games options, one of 5x5 and one of 10x10 as it was asked in the task, the user will need to select one of these 2 documents in order to begin the game.

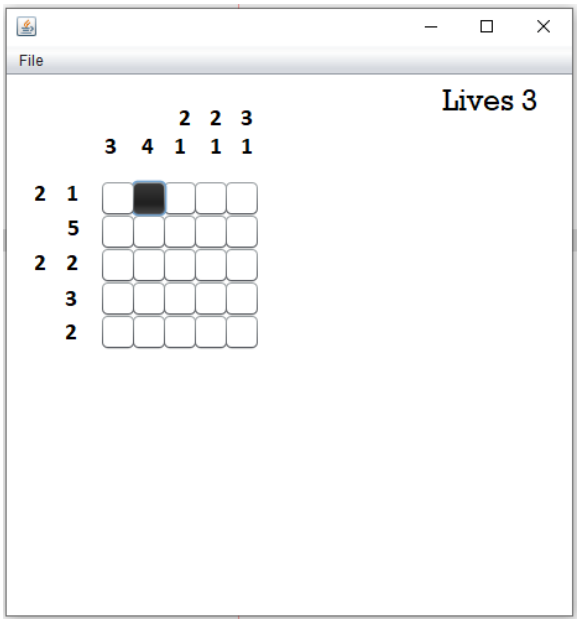
Image 3.

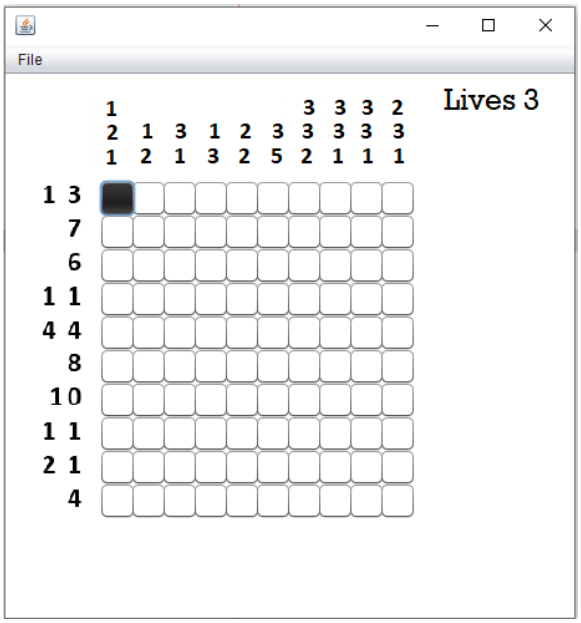




According to the user's selection, he'll see one of these 2 windows: the 5x5 nonogram game or the 10x10 nonogram game. In this moment is where the user can start playing the game.

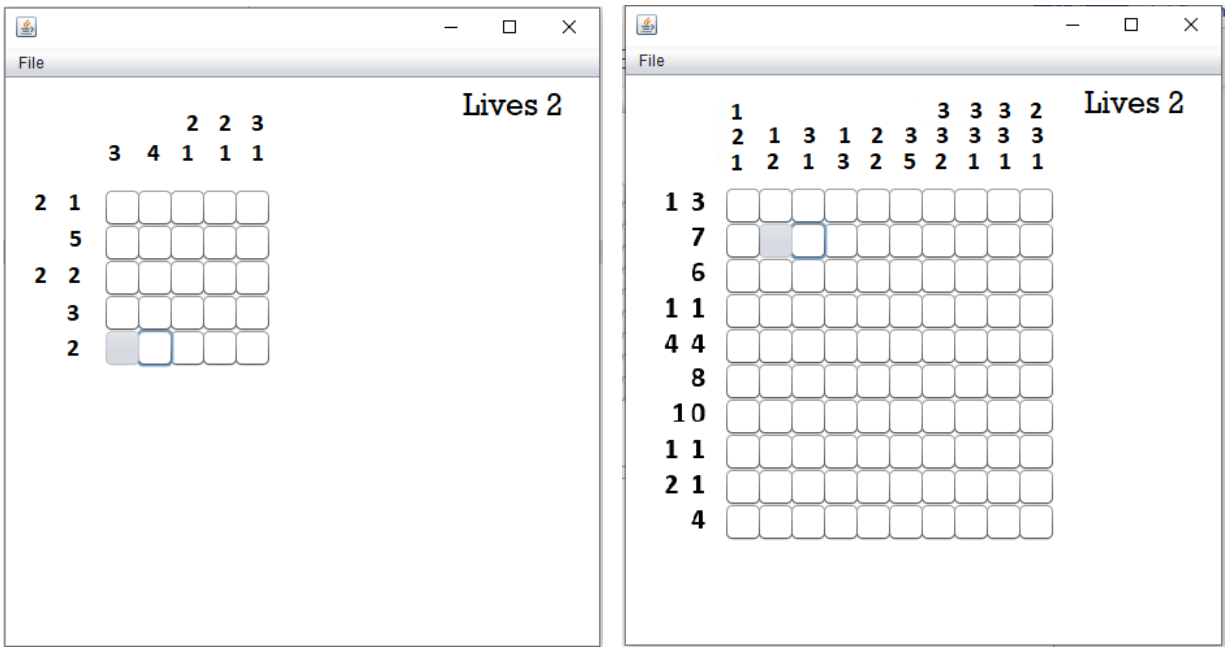
Image 4.





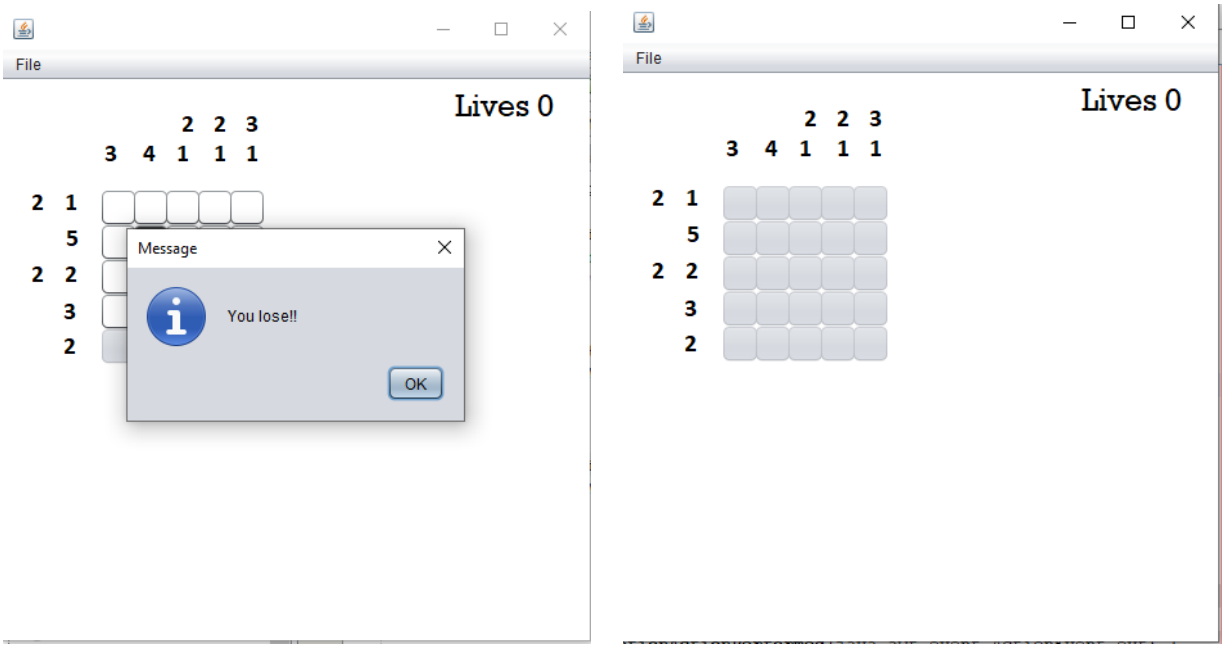
In order to play, the user needs to click any of the non-blocked cells (he needs to click the ones that correctly solves the combination presented/pre-charged by the nonogram). If the user clicks on a correct cell (a cell that must be selected in order to win the game), the cell's color will change to black, and the lives counter of the game will not decrease.

Image 5.



If the user clicks on one of the incorrect cells (a cell that must **NOT** be selected in order to win the game), the cell will get blocked and the lives counter of the game will reduce its number by 1.

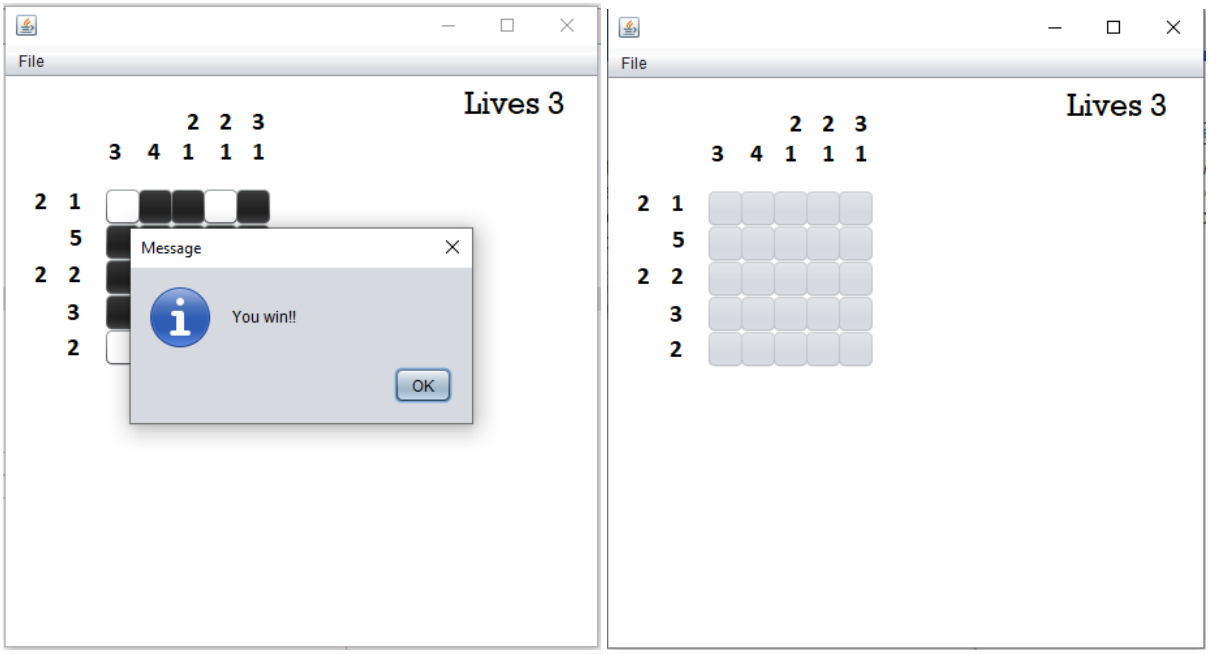
Image 6.



If the user clicks on 3 incorrect cells, his lives counter will get to 0, which means that he has lost the game. When this happens, a message saying: “You lose!!” will appears on

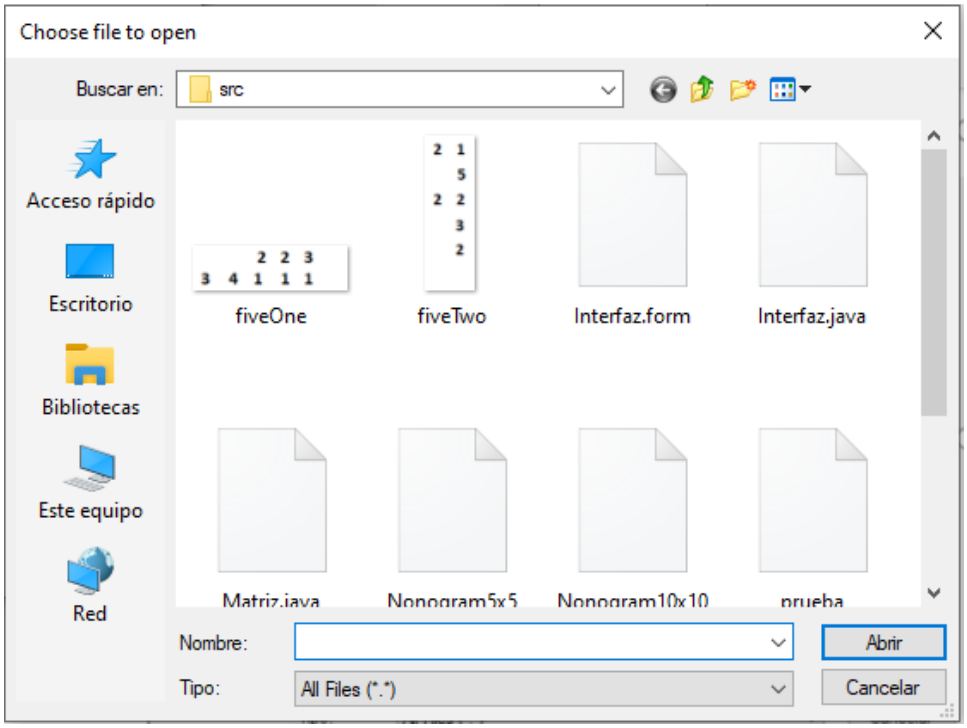
the screen. After the user clicks on OK button, all the cells will get blocked and the user won't be able to continue with the game.

Image 7.



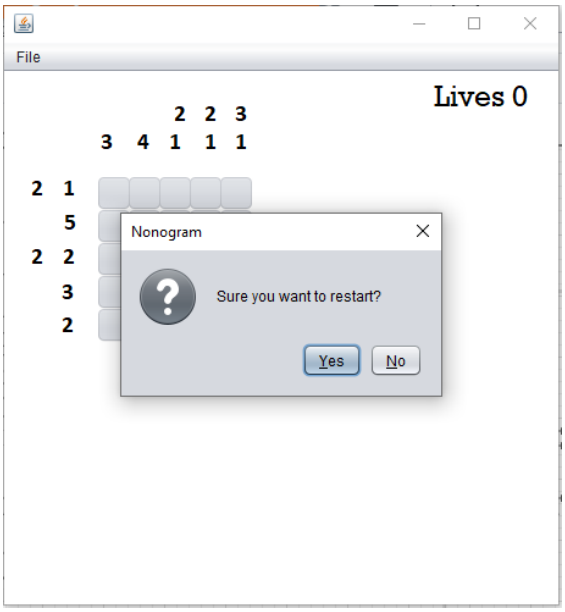
If the user clicks on all the correct cells, remaining at least with one live on its live counter, a message saying: "You win!!" will appears on the screen, which means that he has won the game. After the user clicks on the OK button, all the cells with get blocked and the user won't be able to continue that game (as it already finished).

Image 8.



If the user clicks on the “Open Game” command, a window of the programs folder will open, where the user will be able to select the document (the game) that he wants to open in order to start playing.

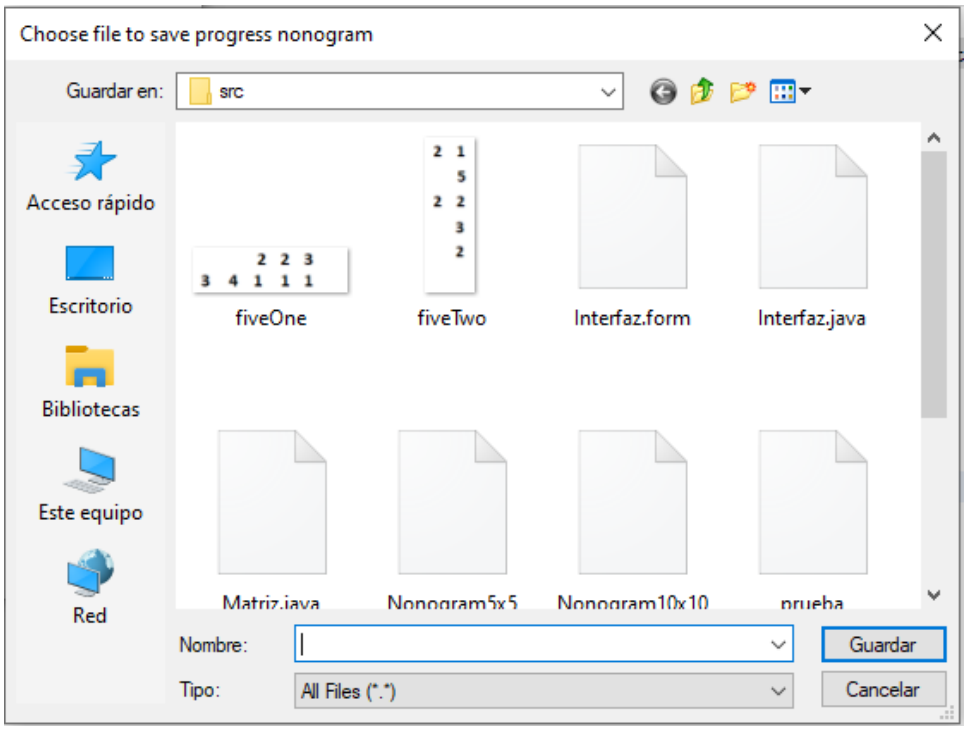
Image 9.



If the user clicks on the “Restart Game” command, a message will appear on the screen asking to the user: “Sure you want to restart?”, if he clicks on the Yes button, the program will restart from scratch the game already previously opened (5x5 or 10x10),

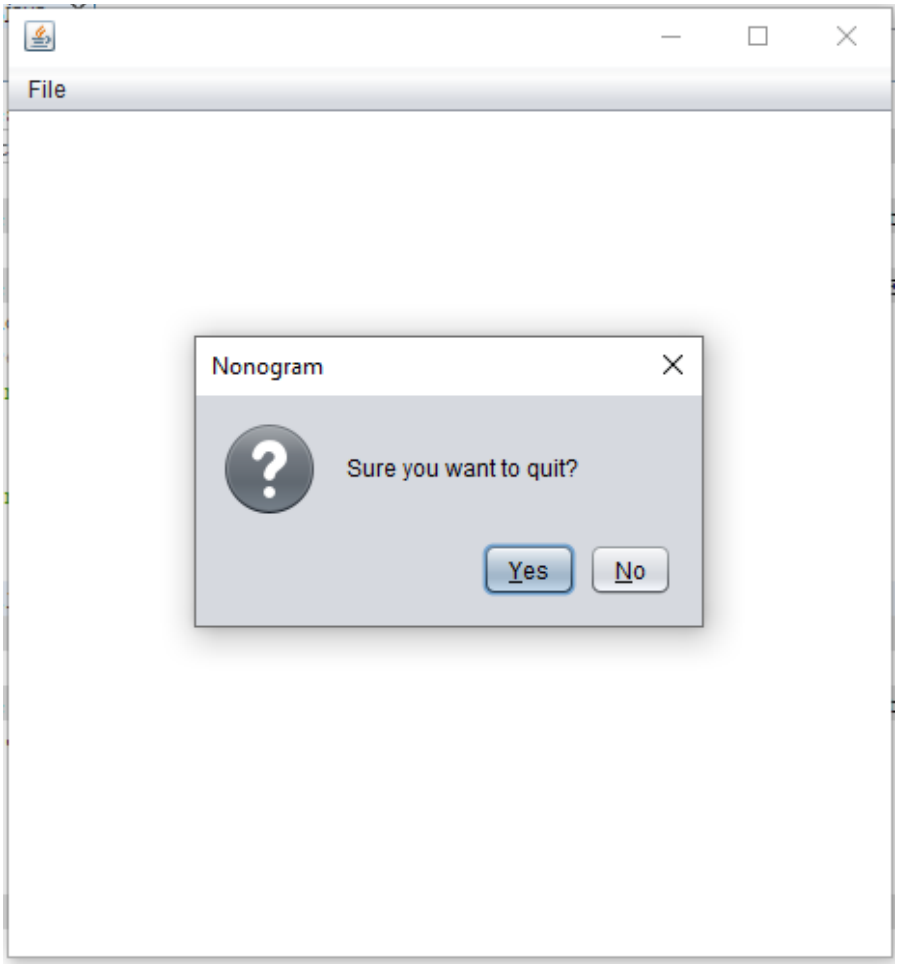
having all the cells unlocked and the lives counter on 3; if he clicks on the No button, he'll just get back to the previous screen.

Image 10.



If the user clicks on the "Save Game" command, a window of the programs folder will open, where the user will be able to name the document (the game) and save it in the folder where it is wanted.

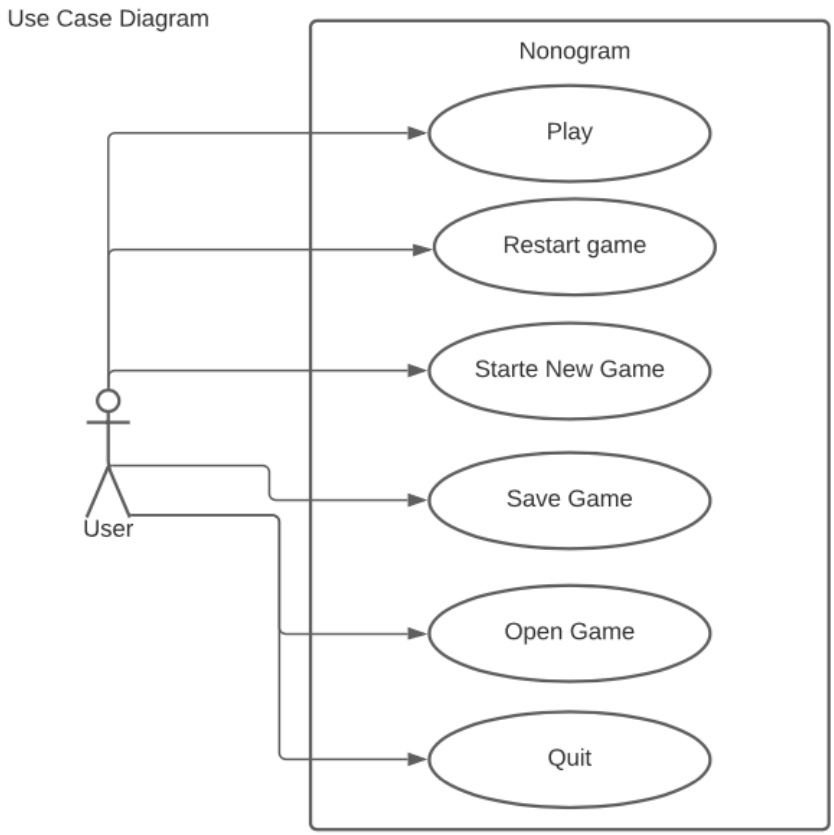
Image 11.



If the user clicks on the “Close Game” command, a message will appear on the screen asking: “Sure you want to quit?” to the user, if he clicks on the Yes button, the program will close immediately; if he clicks on the No button, he’ll just get back to the previous screen.

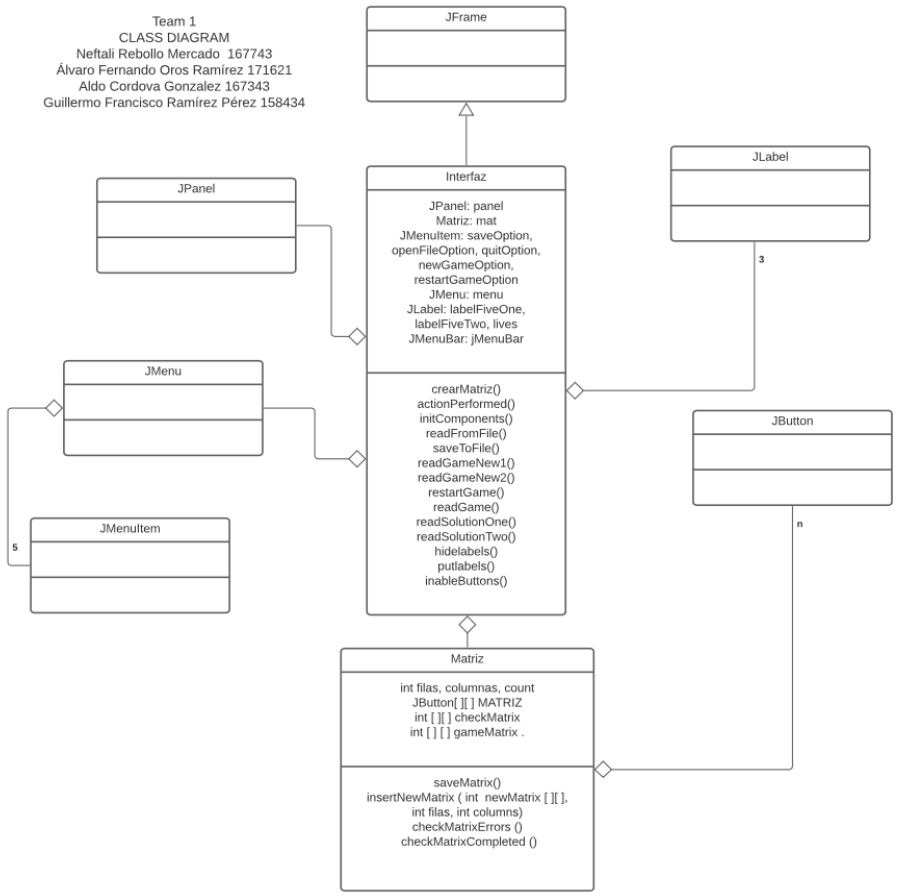
Diagrams

Image 12.



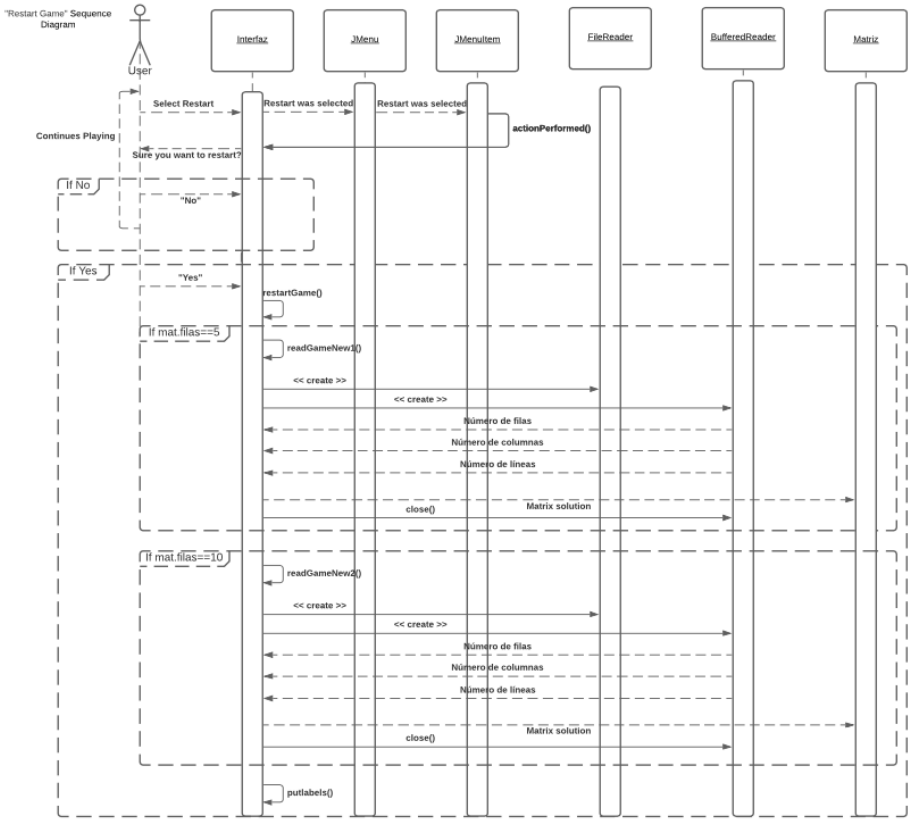
In the Use Case Diagram, we can see all the process/commands that the user can do in the program: play (the whole process of playing the nonogram), restart game, start new game, save game, open game and quit game.

Image 13.



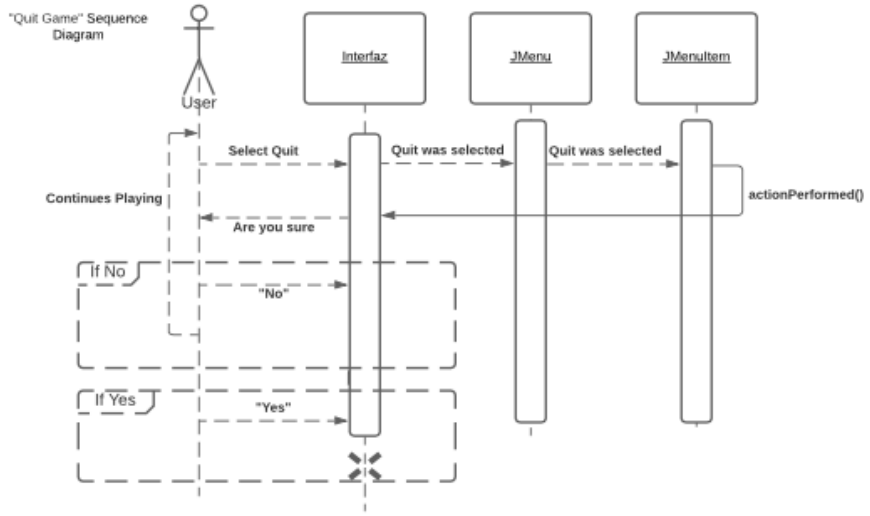
The main class of the Program is called “Interfaz”, this class inherits from the superclass JFrame. This class has as attributes JPanel panel and an object of the class Matriz “mat”. Matriz, JMenu , JLabel and JPanel classes have an aggregation relation with the main class. This class has the methods crearMatriz(), actionPerformed(), initComponents(), readFromFile(), readGameNew1(), readGameNew2(), restartGame(), readGame(), readSolutionOne(), readSolutionTwo(), hidelabels(), putlabels(), saveToFile() and enableButtons(). Also, we have five JMenuItems: saveOption, openFileOption, newGameOption ,quitOption and restarGameOption. The diagram indicates the instances that will be of the JButton class will be “n” number of buttons and we will only have one of the class Matriz. Also, the JButton class is related with the class “Matriz”. Finally, we can see in the class “Matriz” the integer variables columns, rows, an array of JButtons “MATRIZ”, and two integer arrays. Also, Matriz has the methods saveMatrix(), insertNewMatrix, checkMatrixErrors () and checkMatrixCompleted().

Image 14.



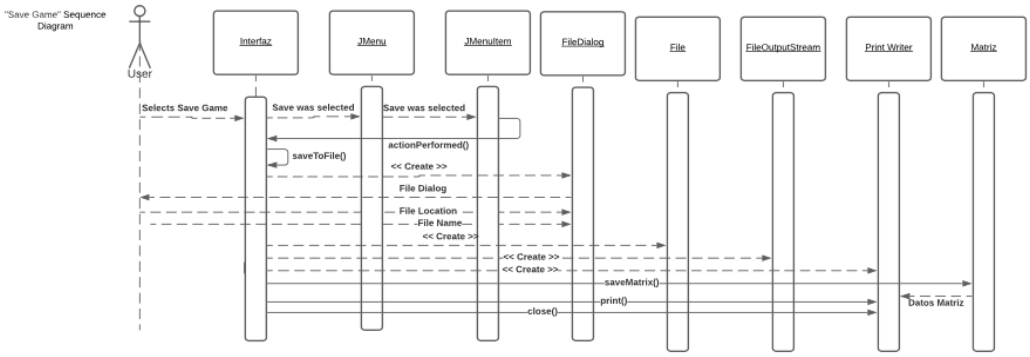
The user selects the option to “Restart Game” so the class Interfaz indicates to JMenu that save was chosen and it tells the same to JMenuItem then method actionPerformed () sends the event to Interfaz. Then, so the Interfaz asks the user “Are you sure to restart?” and she/he answers. After, the method restartGame() starts, it removes the componentes and create a new matrix of buttons. There are two ifs that determine the size of the matrix chosen to play, depending on the size the program create an object of FileReader and BufferedReader then it creates the matrix. Finally, Interfaz receives the lines, rows and columns of Buffered Reader and sends them to a matrix in the class Matriz. Finally, the Interfaz calls the method close() from Buffered Redear and also it calls the method putlabels().

Image 15.



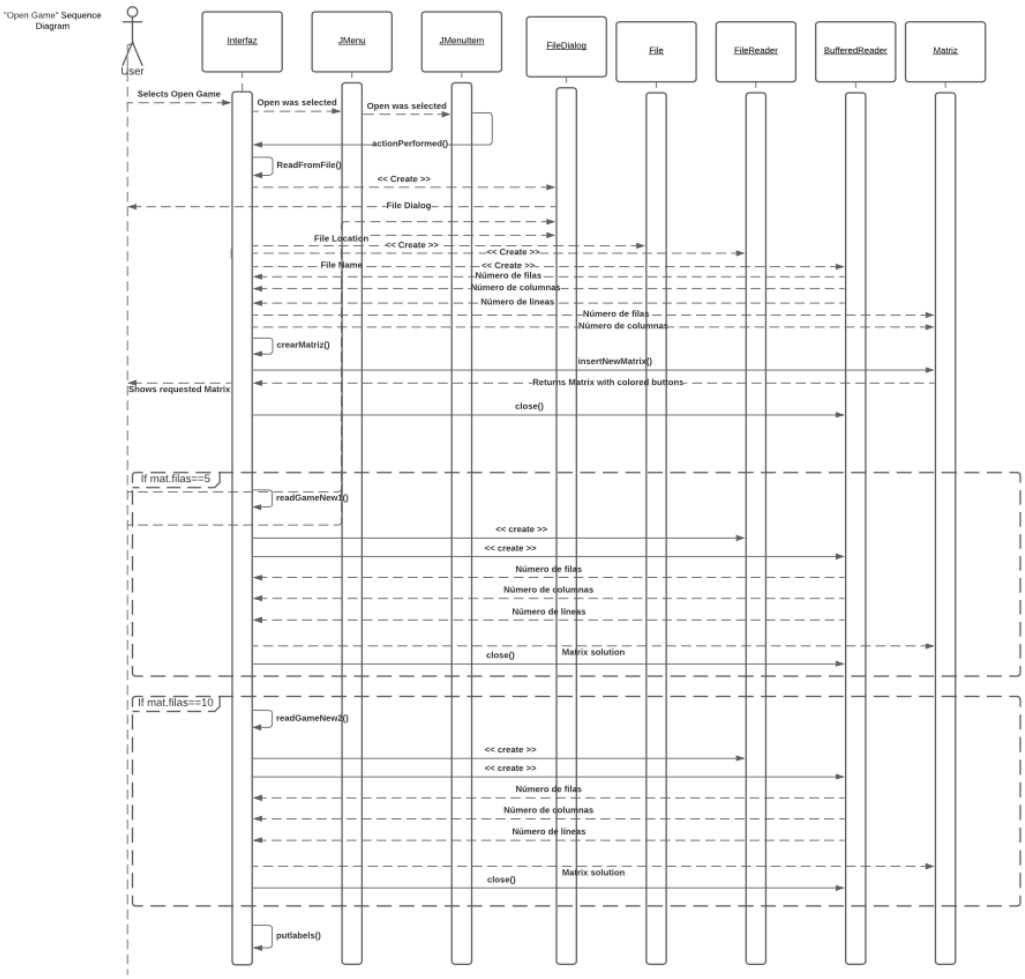
In the Quit Game sequence Diagram, the user selects from the Interfaz the option “Close Game”, then the Interfaz communicates to with the JMenu and the JMenu with the JMenuItem, then the JMenuItem calls the method actionPerformed() so the Interfaz can display a window, which will ask the user if he/she wants to quit, if “No”, then the user can continue playing, but if “Yes”, then the program (Intefaz) stops.

Image 16.



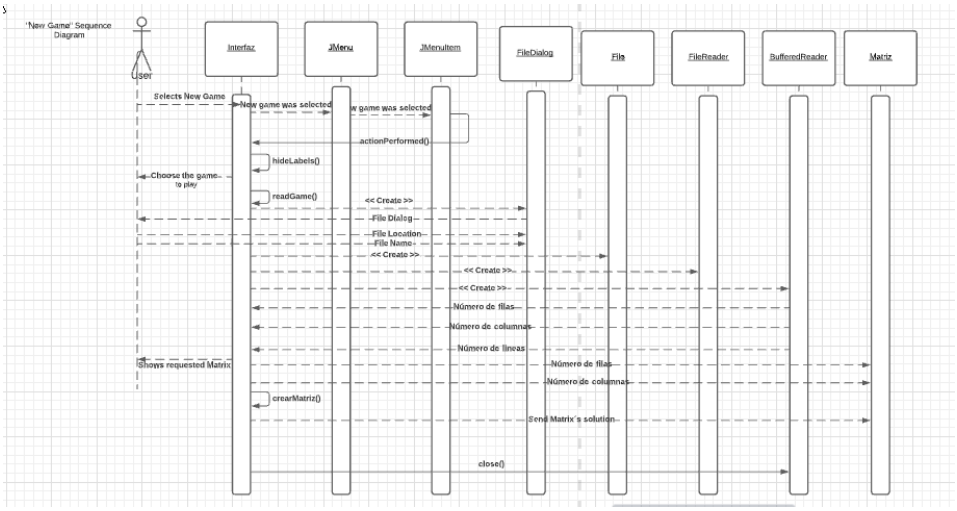
The user selects the option to “Save Game” so the class Interfaz indicates to JMenu that save was chosen and it tells the same to JMenuItem. The method actionPerformed () sends the event to Interfaz and it starts the method saveToFile(), to save the matrix into a file. The Interfaz creates an object of the class FileDialog and it shows a window to the user so he/she can decide the name and path to save the file and this data returns to FileDialog. Then, it creates an object of the class File and PrintWriter. After, the Interfaz calls the method saveMatrix() of Matriz and it returns the information to PrintWriter and it closes the PrintWriter.

Image 17.



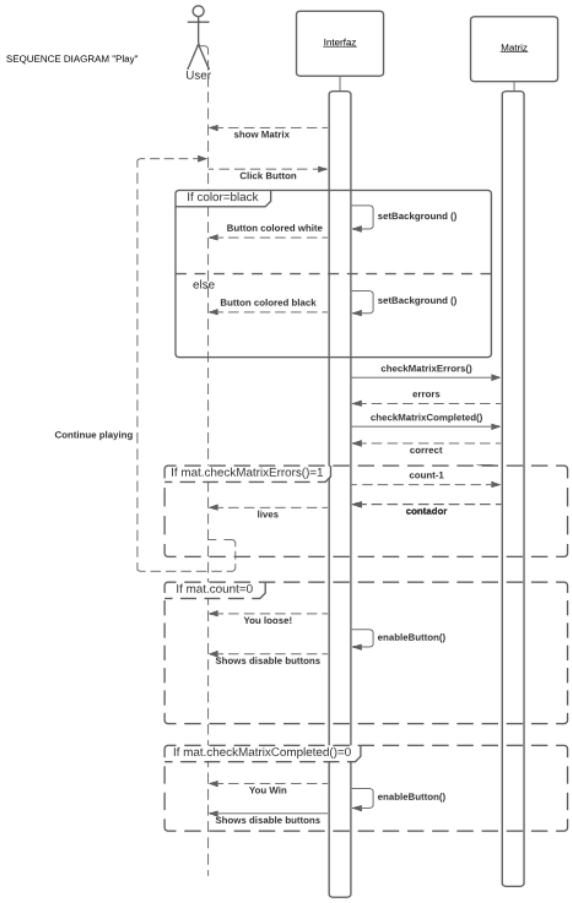
The user selects the menu Item “Open game”, the Interfaz notifies JMenuItem that it was clicked and the method actionPerformed() starts. Then, the method readFromFile() begins and an object of the class FileDialog is created. The FileDialog displays a window so the user can select a file and he/she returns the path and the name of file. Later, an object of File and BufferedReader are created. The BufferedReader reads the rows, columns and lines of the matrix and passes them to Interfaz. It sends the rows and columns to the class Matriz and the method crearMatriz() assigns the color and actionListener to each button. Interfaz calls the method insertNewMatriz() of the class Matriz() to color the JButton matrix and receives the matrix with colors. At the end, the class Interaz calls the method close() to terminate BufferedReader. Then,with two ifs the program reads the solution depending on the size of the matrix.

Image 18.



The user selects the menu Item “New game”, the Interfaz notifies JMenuItem that it was clicked and the method actionPerformed() starts. The class Interfaz calls the method hidelabels() to hide the labels around the board and send the message to choose the game to play to the user. Then, the method readGame() begins and an object of the class FileDialog is created. The FileDialog displays a window so the user can select a file and he/she returns the path and the name of file. Later, an object of File and BufferedReader are created. The BufferedReader reads the rows, columns and lines of the matrix and passes them to Interfaz. It sends the rows and columns to the class Matriz and the method crearMatriz() assigns the color and actionListener to each button. Interfaz sends the solution of the game of the class Matriz(). At the end, the class Interaz calls the method close() to terminate BufferedReader.

Image 19.



When the user wants to play the class Interfaz show us the Matrix and the user click button. If the color is black and it´s clicked it returns to white and when the button is white it changes to black. Everytime, the user clicks a button the Interfaz calls the method checkMatrixErrors() of the class Matriz and receives the number of errors. Also the method chechMatrixCompleted() is called and it returns the value correct. If the mat.count gets equal to 0, it means that the user has lost his 3 lives and he has lost the game, so interfaz will show a message to the user saying: “You lose!!” and disable all the game buttons. Finally, if at a certain point the mat.checkMatrixCompleted() is equal to 0, the interfaz will show a message to the user saying: “You win!!” and disable all the game buttons.

Conclusion

With this project, we were able to put into practice all the programming skills that we learn throughout the Object-oriented programming. In general terms, the assignment we were given was to create a nonogram with at least 2 different games to play, and some basic program functions (new game, open game, save game, restart game and close game). As a team, we manage to delegate the different tasks needed to complete the

assignment so we could work on them separately but reviewing them all together to improve the work done.

At the end, we create a nonogram program that have 2 different game sizes (5x5 and 10x10), with a lives counter, with all the basic program functions required and that have been prooved by different people without showing any error and running appropiately