Game Box

CSE1142_Project

Authors

150119864 Abdulaziz Alftaieh

150119632 Beyzanur Çabuk

CSE1142 Computer Programming II, Spring 2021

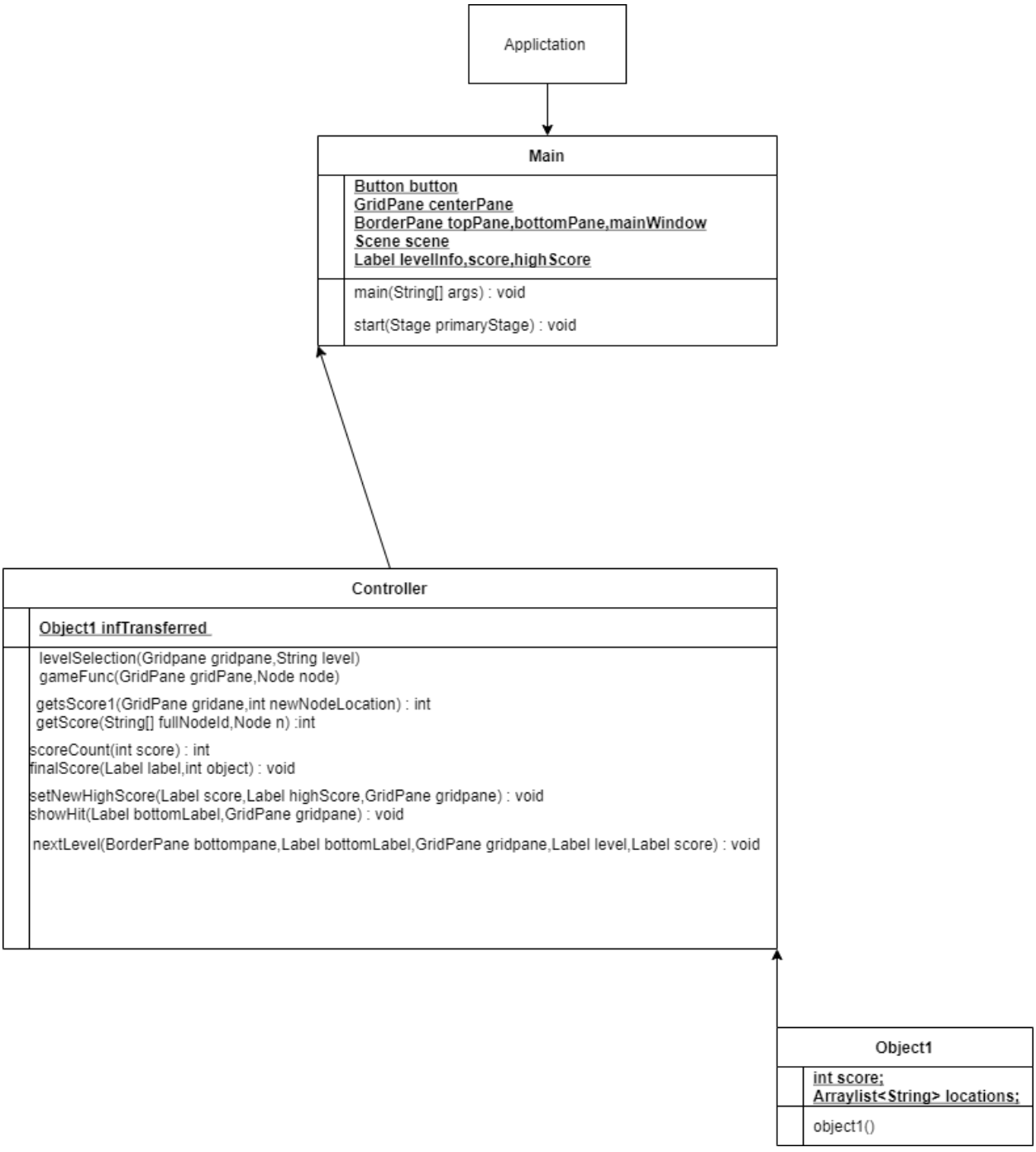
May 28, 2021

Problem definition

Game box is game where the user has to click on boxes until they are all empty or generic one (Wall type) at first they will see a user interface that includes four labels and 100 boxes ad each box can be either grey, white, cyan or hot pink and then they can click on any of the boxes but there are some problems that will be faced one of them is to check for the clicked box (clicked button) is it grey (Wall) or white (Empty) or cyan (Mirror) or hot pink (Wood) so after checking which color is it there has to be other checking for it's left side, right side, upper side, and the box (Button) beneath it and the same process of checking has to be done for each and every one of them after that we have to calculate what is the total score of the user click and surely there is no sequence for the scoring system that we have so they have to put individually.

So after finishing up the score calculation in the back end we have to add each and every box that was hit so that we can give the user the opportunity to see what are the results of his click for example if he hit box 8,1 then we have to first set the new color if the clicked box then search for all the other boxes which are 4 other ones and then include the ones who were hit and change their color according to the rules that we have got in the pdf file after that we have to update the bottom label with what we have got from the click and that again would take so much work and more code and after that we have to update the score by adding the previous score to the new obtained score so we need to change the text and parse it to an integer then add the new calculated score and then check if the score is higher than the high score we update it to be the

new high score soon after that we look if all the boxes that were a mirror type or wood type to be empty and so if they were we will create a button that says next level to the user and read the next level text file to import the boxes where we need to fill and be filled according to the text files then we increase the level label by one of course and zero the high score after that we make the score zero because it’s a new level and set back the bottom label to the generic ---Text--- after that we delete the new added button for the next level and this process repeats itself smoothly till the user reaches level five where maybe he doesn’t know that he finished all the levels and there are no more levels to finish so the bottom label gets updated that finished all of the levels and by that the game finishes.



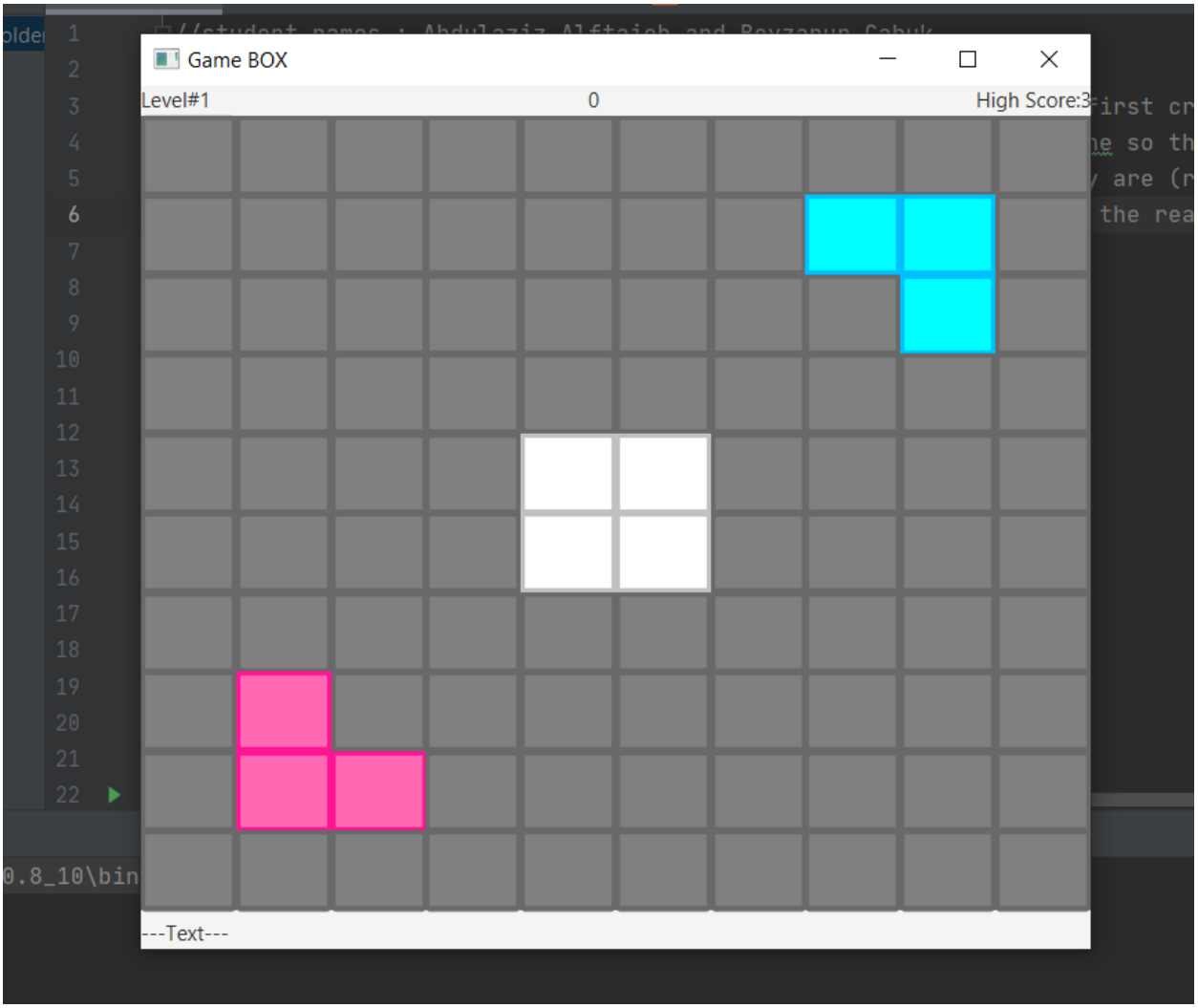
So this UML diagram shows how the flow of the program is first because it's a javafx project we have to inherit the Applications class then we have the main class which is mainly about what the user can see and how the buttons are and their style and we used three border panes first one is to include all the panes together then the top pane which made make the three labels created in it stay in their right positions even if the user decided to make the screen bigger or smaller and

same for the bottom pane where we have made a bottom label that keeps getting updated with every hit and informs the users if the level text file is missing and then for the buttons they were created through a nested loop 10x10 just as intended and every button was set with a certain id that holds their row and column and their type and their position in a 1D array and lastly their type but as in a number to be easily debugged and tracible where 0 meas a Wall type and 1 = Empty and 2 = Mirror and 3 = Wood and then an Event Handler was made so that if the user clicks on any of these button they would react in the same way first we run the gameFunc which is a shortcut for gameFunction and it takes two parameters the GridPane (centerPane) and the node that was clicked and we used the .getSource function in the Actionevent class so that we know from which node did the click come from then we used the finalScore to check the final score from that hit then the showHit method to update the bottom label with every click from the user and then we cleared the Arraylist in locations object so that we don't mix up the old locations with the new ones plus it would be better for the data consumption then we initiate the setNewhighScore method but it will work only if the score is more than the high score in the top right of the screen lastly we check all the boxes if they are either empty and walls or not and if they were we initiate the next level of course with keeping in mind that maybe the text file isn't there or the user has finished all 5 levels so a text appears in the bottom left of the window indicating that they either finished or there is problem with going to the next level b by using the exception file not found now for the Controller class this is where all the logic and backend is where at first we initiate our custom made object to transfer the information that we need and that was a problem at first so we came with solution to create this custom made class and call the object infoTransferred and it has two fields first an integer to hold the score and send it to the main window and the second is an array list of strings that will hold the 1D positions of the hit boxes the clicked box to later on be used in updating the bottom label now after that we made a little trick where we used the .format method from to String class to have the same directory but the number of the level differs based on the parameters then we check each line and see if they match with the id of a button and when they do simply just take the button and change it to the intended style after that we close the scanner and we are done with the levelSelection method then we go the gameFunc method where it takes two parameters first a GridPane and the Node that has been clicked so after that we thought about how we can make certain if conditions so that we don't write a lot of code so we at the end just came with simple idea to treat them like a

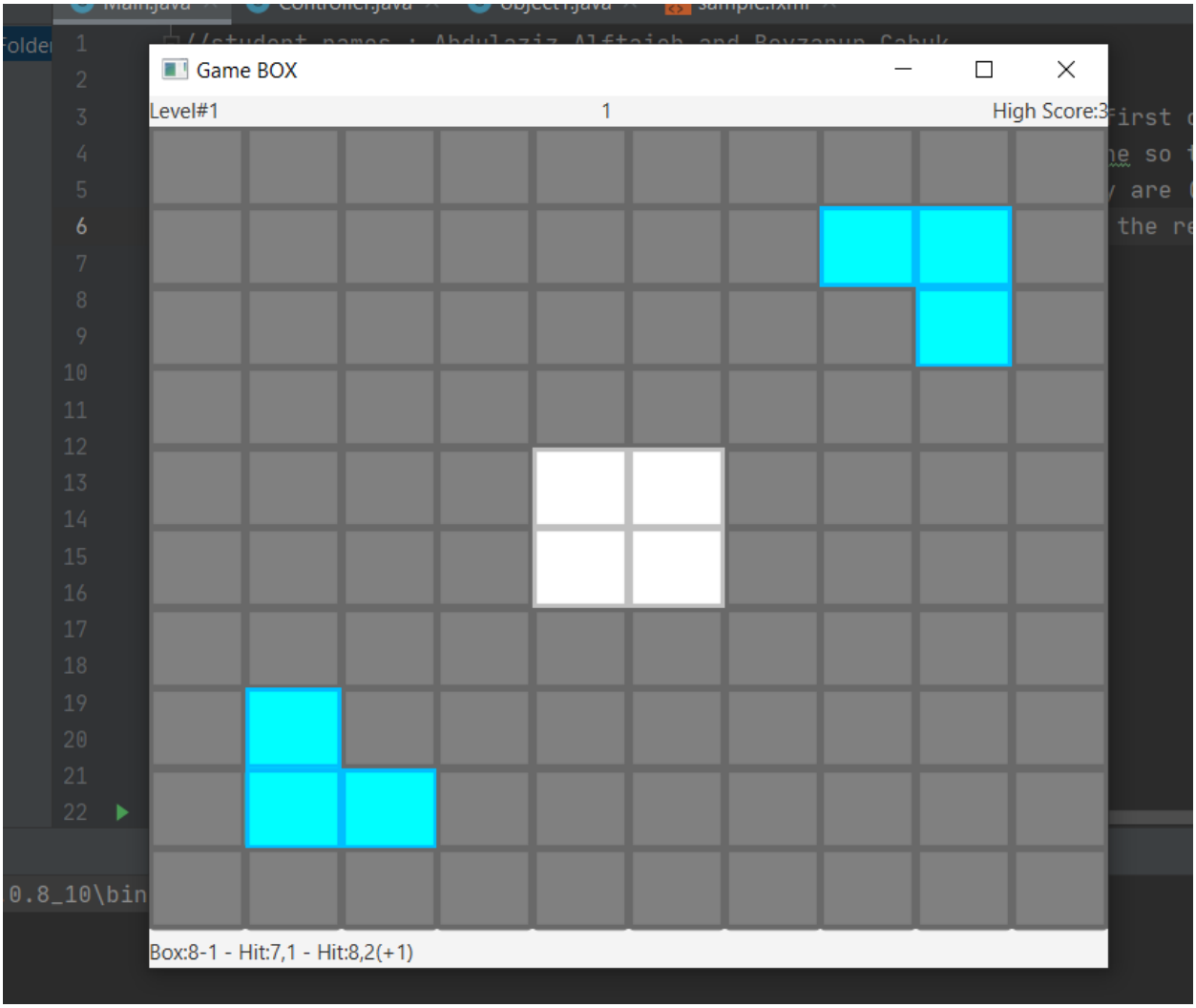
matrix and started to write the if conditions first there are three conditions if they were Wall or Empty just do nothing second if the clicked box (button) was a Wood box (button) then we have to make 4 exceptions the first one if the box location was zero then we have to check for the position next of it and the one beneath then the other one is if the position was 9 then we have to check the one before it (Left side) and the one beneath it or the 19th box and the 3rd exception was if it was 90 we have to check the upper one and the one on the right by adding one for the right one and minus 10 for the one above and the last exception is if it was 99 then we check the left side and the upper side and then we check the ones between 1 and 8 because they can only have sides same for the ones between 0 and 90 and 91 to 98 but for the right side of the grid we had to do them manually by making 19 or 29 or 39 or 49 etc.. till or 89 and what we do for each box is that first we check if they are wood or mirror boxes and if they are we initiate the getScore method and this method checks each potential node that was hit from the box clicked by initiating the getScore method were they get their new colors and it returns 1 and that means that use this node was actually it form the clicked box and then add their 1D position to the infotrnasferred.location array list so that later on it can be used to for the bottom label to show what was hit and the scoreCount method just takes the score takes the overall hit boxes including the clicked box and adds them all up and checks how many hits then we assign it to the infoTransferred.score to later on use it to update the score label in the game window and then the final score takes two parameters the old score and adds it to new score entered so that we can show it on the screen to the user for the setNewHighScore method it runs a loop through the centerPane (GridPane) and then checks if all of the boxes are either empty or not Wall and if they were it updates the high score label on the top right of the game window and for the showHit method it has a nested loop firstly looping through the infTransferred.loctation array list and the second loop is through every box in the gridpane and first it takes the first hit as a box: and the box(button) location and then it keeps adding in the string every confirmed hit made by the box after that it updates the bottom label with the whole string and for the last method we have nextLevel method where we check if all of the buttons are either empty or a wall tye and if they were we create a button out side of the loop and set the text to Next Level >> and if the user presses that button first we check if he/she has reached level 5 and we update the bottom label with a message informing the user that they finished all of the levels or if the next level text file

was missing we threw an exception and update the bottom label informing them that the next level text file is missing .

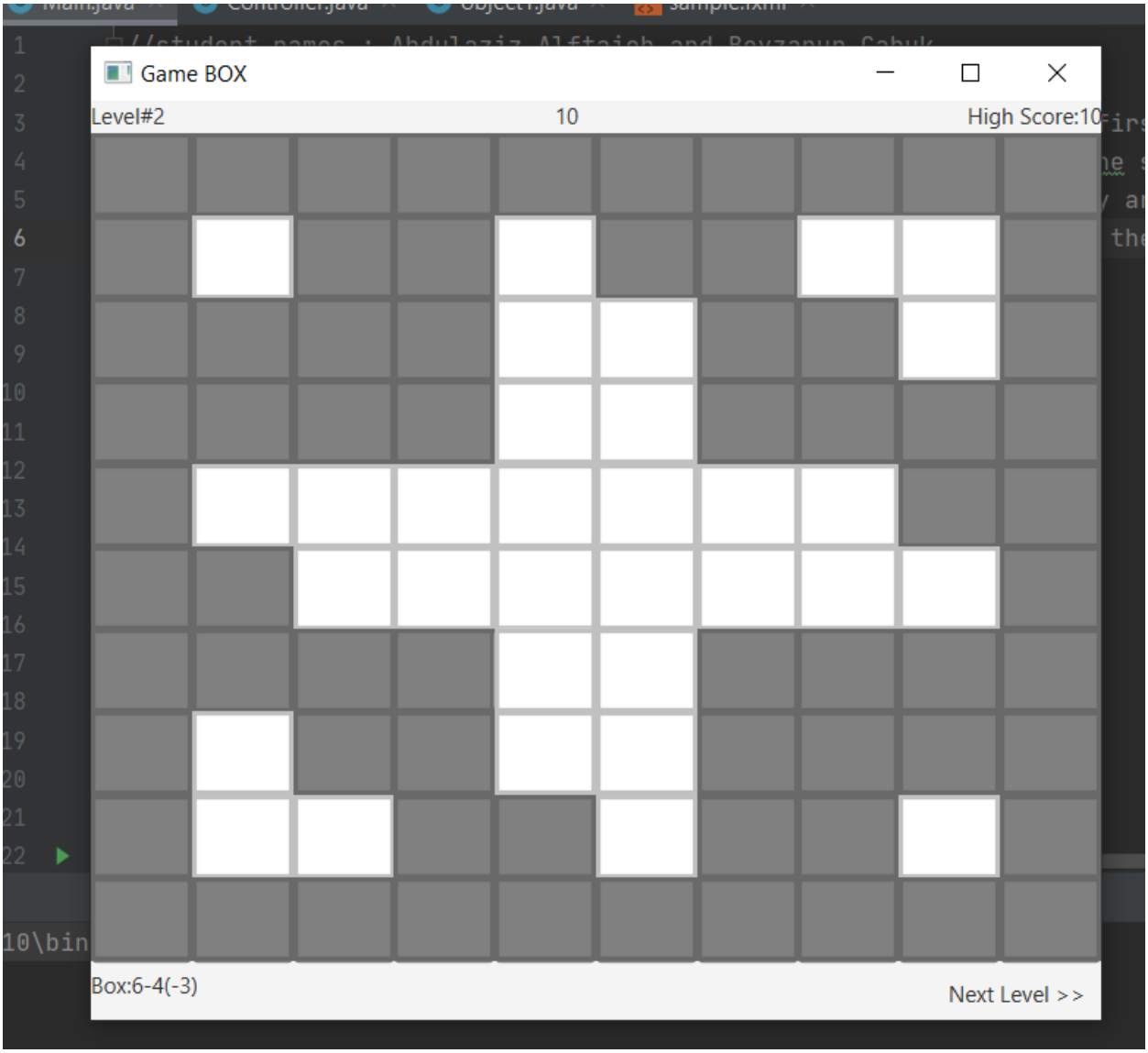
Test cases



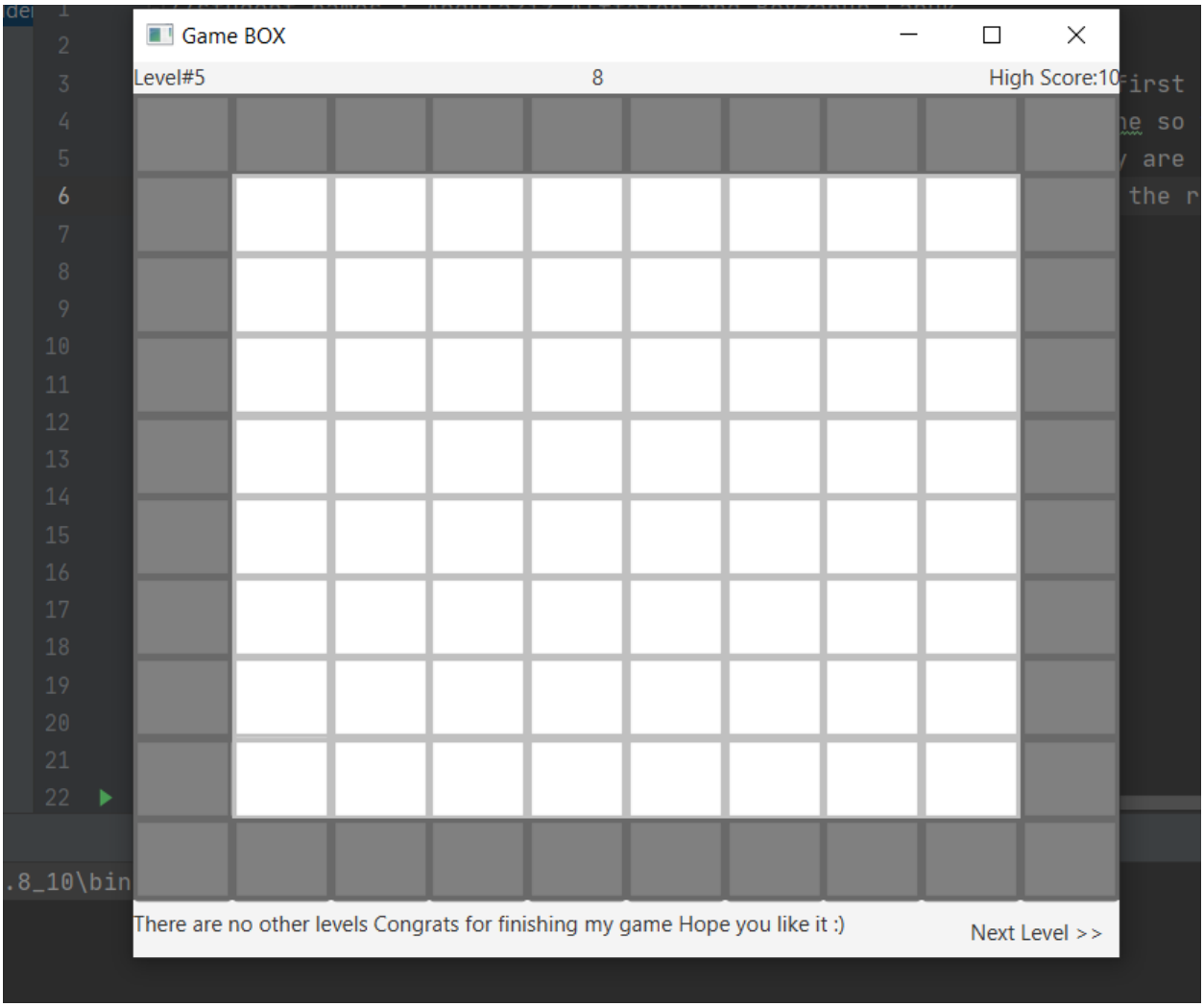
now as you can see here this is what pops up when you first open the window starting by the level 1 and the score is and the high score is 3



And the hitting came correct with the correct text in the bottom label



When we finished level 2 the next level was shown and the high score was updated



When we reached level 5 and finished it and clicked the next level button it was disabled as intended and a text popped up saying that are no more levels to show up