

**MARMARA UNIVERSITY**

**FACULTY OF ENGINEERING**

**COMPUTER PROGRAMMING II    TERM PROJECT**

**CSE 1242.1   Spring 2022**

**Date Submitted: May 09, 2022**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Dept** | **Student Id** | **Name and Surname** | **Contribution** |
| **1** | **CSE** | **150120043** | **Umut BAYAR** | **Code Design, Report Preparing** |
| **2** | **CSE** | **150121004** | **Ahmet Arda Nalbant** | **Code Design, Report Preparing** |

**PROBLEM DEFINITION**

The name of our project is pipe puzzle. This game contains 6 different levels.Each level is a little more complex than the previous level.  The purpose of this game is to connect the pipes to provide that the ball moves.The game has some specific rules. For example, if you are in the first level and can’t solve it yet, you cannot play other levels. Another important rule is that tiles cannot move diagonally.There are 7 pipes that are different from each other. These are ;

Starter;  These pipes do not move but can be found horizontally or vertically and There is only one starting tile per level. End pipes are static. End tiles can be vertical and horizontal. The end tile is the pipe where the ball finishes to slide at each level and there is one at each level.Empty Free; These tiles cannot move but other tiles can move the position of this tile. Empty tiles are tiles without pipes. Empty; These tiles cannot be replaced by other tiles, but they can move to the positions of empty tiles.  Empty tiles are tiles without pipes. Empty tiles are tiles without pipes. Pipe Static; Their locations are static and they can be horizontal or vertical.Pipe static tiles cannot move. Pipe;  They can be horizontal or vertical and moved to the position of empty free tiles. It is the most common type of pipe in the game. Curved Pipe; Curved pipes can be placed in an empty tile position. Curved pipes are available in four different shapes. Different names are given in terms of shape properties.

We chose an interesting background photo for our game. In addition , we have added level buttons to see the levels and start from the level we want.  There is a welcome text to the pipe puzzle game at the top. Then after choosing the level and starting the game, our game starts. We have to start the game from the first level. We cannot go to the next level without finishing a level.Also when we complete the game , the message “You won” appears on our screen. There is also a counter that shows the number of our movements.

**IMPLEMENTATION DETAILS**

1. **UML diagram of the Game Class**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  | Game |  |  |
|  |  |  |  |  |
|  | + | id: int | | | |
|  | + | type: String | | | |
|  | + | property: String | | | |
|  | + | board: Game[][] | | | |
|  | + | levelUnlocked: int | | | |
|  | + | path: Path | | | |
|  | + | Game() | | | |
|  | + | Game(int id, String type, String property) | | | |
|  | + | isLegal(int x0,int y0,int x1,int y1): boolean | | | |
|  | + | generateMapData(String inputFilePath): void | | | |
|  | + | makeMove(int x0,int y0,int x1,int y1): void | | | |
|  | + | gameEnded(): boolean | | | |
|  |  |  | | | |

* This class has the operation mechanism of the game, data retrival and animation.
* Each box has an id, type and property. (7,Empty,none)
* The variable board is a multi-array and stores the value of the x and y.
* The variable levelUnlocked is int and stores last unlocked level.
* The variable path is used for finding right path to end the game.
* Game constructor’s parameter is empty. It is used for setting board to 4 by 4, levelUnlocked to 0 and path.
* Game constructor’s parameter is int id, String type, String property.
* isLegal is a method and it’s parameter is int x0,int y0,int x1,int y1 and return type is boolean. This methods checks the move is legal or not. If start tile’s type is Starter, End, PipeStatic or Free it returns false. If the target tile’s type is Free and the distance is 1 return true.
* This method called generateMapData generates the mao of the game. read’s type is BuffrederReader and it reads level from inputFilePath. It holds id, type and property to place these values in specified rows and columns.
* This method called makeMove swaps the coordinates of the tiles.( int x0,int y0,int x1,int y1)
* This method called gameEnded is boolean type and checks the game is ended or not. This method have a newCell array. It stores new cell to get type and property. With the help of the enum it detectes where the ball must come adn where to go and according to that animation is also created.

1. **UML diagram of the Project Class**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  | Project |  |  |
|  |  |  |  |  |
|  | + | game: Game | | | |
|  | + | level: int | | | |
|  | + | numberOfMoves: int | | | |
|  | + | data: String[] | | | |
|  | + | boardGrid: Rectangle[][] | | | |
|  | + | panel: GridPane | | | |
|  | + | gameStatus: Label | | | |
|  | + | countMove: Label | | | |
|  | + | ball:Circle | | | |
|  | + | welcomeScene: Scene | | | |
|  | + | gameScene: Scene | | | |
|  | + | gameEnded(): boolean | | | |
|  | + | animation: PathTransition | | | |
|  | + | start(Stage window): void | | | |
|  | - | nextLevel(): void | | | |
|  | - | generateGridPane(): void | | | |
|  | - | setMove(Rectangle tile): void | | | |
|  | - | setPlace(Rectangle tile): void | | | |
|  | - | changeRectangles(int x0,int y0,int x1,int y1): void | | | |
|  | + | playAnimation(): PathTransition | | | |
|  | - | saveGame(): void | | | |
|  | - | loadGame(): void | | | |
|  |  |  | | | |

**QUESTIONS**

1. Which parts are complete/incomplete in your project?

+The first of the situations that we could not complete in our project, when the level is completed and we move to the new level, the ball stays where it left off instead of returning to the beginning of the game. Another problem that we tried hard to solve but could not solve is that when we go back to the main menu in the game, the text that we have completed the previous levels(Completed) does not appear. We have completed the rest of it.

2. What are the difficulties you have encountered during the implementation?

+The first issue we had difficulty with was the animation part. Our animation repeats according to the number of levels as the number of levels increases. We were unable to resolve this error. Apart from that, we had difficulties in keeping the number of levels and formatting the menu accordingly, and we could not achieve this completely. We took care of the other parts and had a lot of fun while doing them.

3. What are the additional functionalities of your project added by your team?

+ We have added an impressive background image to the game. In addition, we have added a menu showing the levels and a welcome text for a newcomer to the game. We also added texts showing the number of moves in the game and the need to try again if they can not complete them.

**TEST CASES**

In this part, we will add the images of our project and the photo descriptions for each level.

The number of moves we need to do to complete these levels and the level text showing that we are in the first level are on the lower left. In addition, if we think that we have completed the game, there is a start button and a return to main menu button to return to the menu.

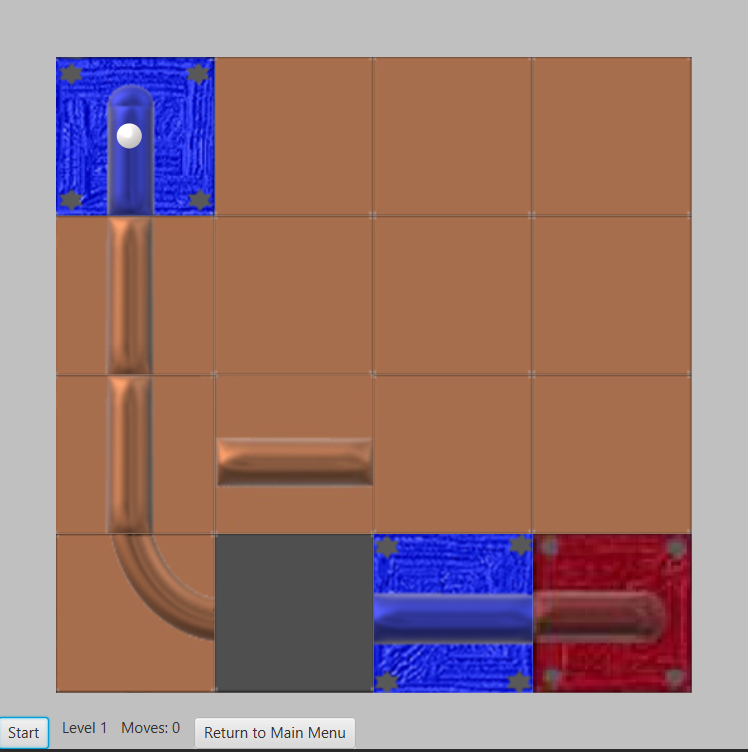
**1-Login Screen**

On this screen , we have our levels and login screen. You can join the game by choosing a level from here.

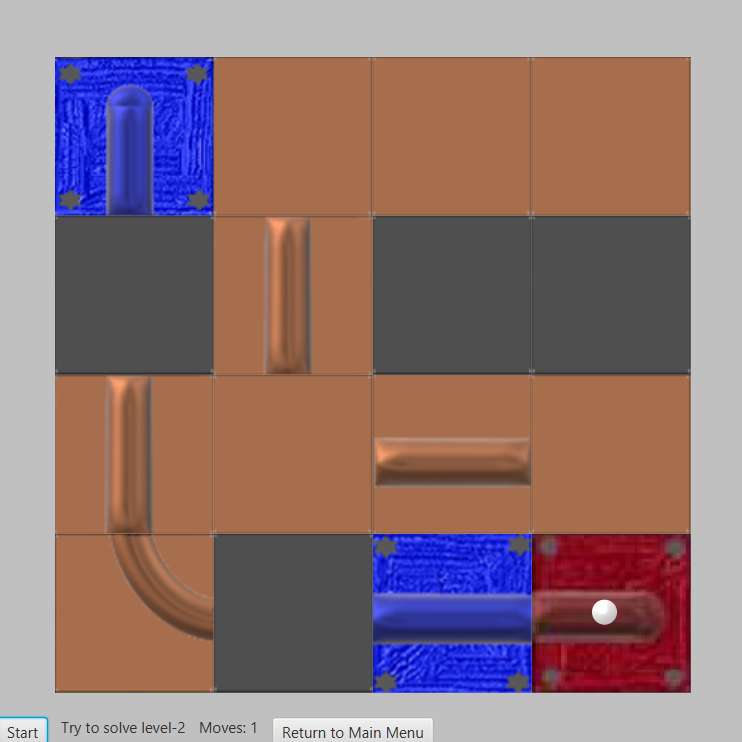
metin, ışık, karanlık içeren bir resim

Açıklama otomatik olarak oluşturuldu

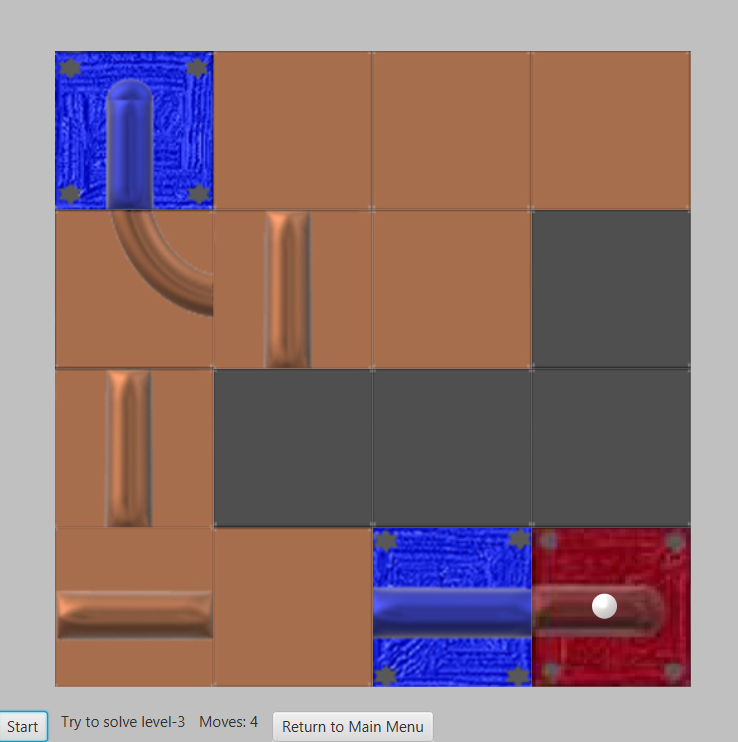
**2-  Level 1**



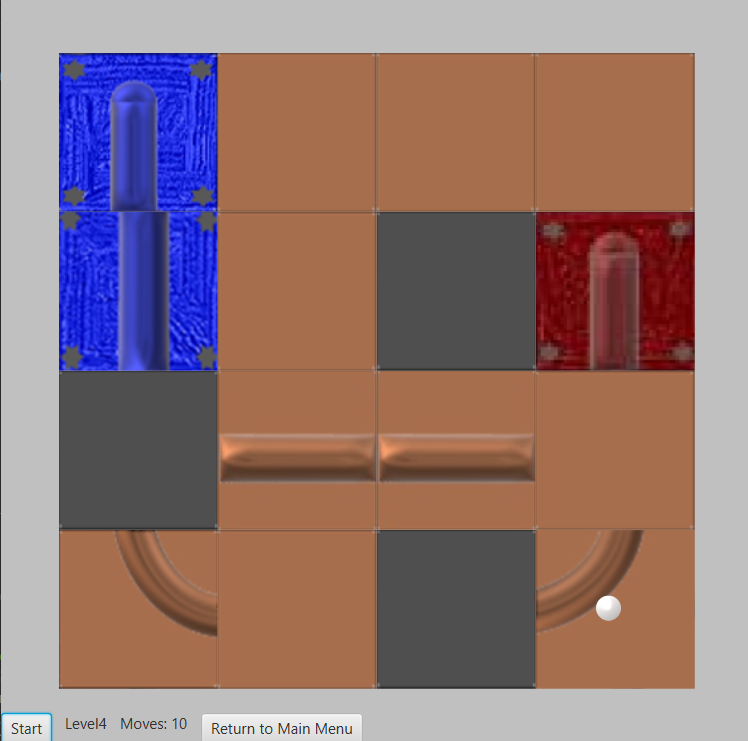
**3- Level 2**



**4- Level 3**



**5- Level 4**

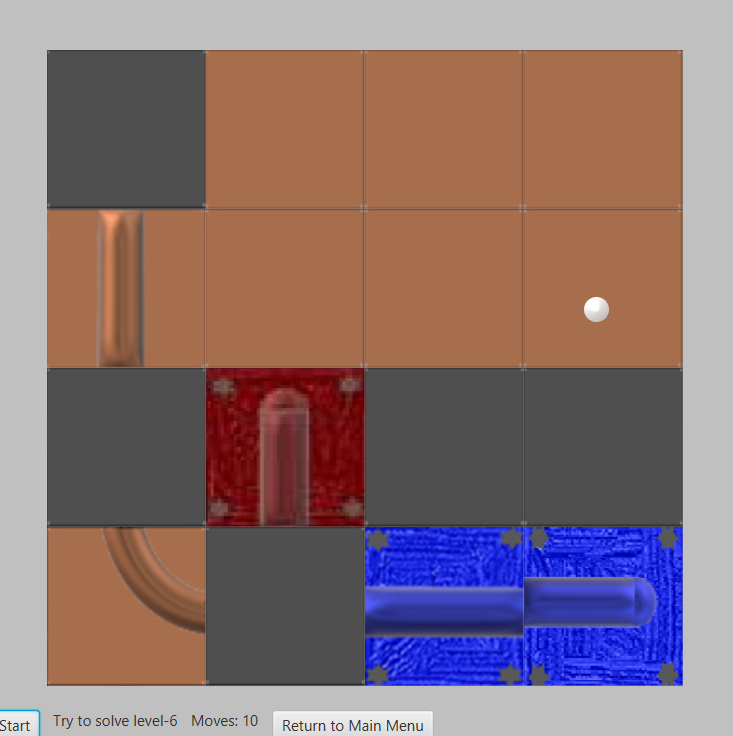


**6- Level 5**

metin, cihaz, mutfak aleti, mikrodalga içeren bir resim

Açıklama otomatik olarak oluşturuldu

**7- Level 6**



**8- Win Screen**

If you successfully complete the game , you will see this screen at the end of the game.

metin, vezir içeren bir resim

Açıklama otomatik olarak oluşturuldu