Alireza Rezaei

Advanced Program Structures Project

City Domino

This document has two parts, user manual and development manual for the game " City Domino".

Third Semester
ITE102772
Alireza Rezaei

Table of Contents

1- User Manual

1-1 Requirements:

The City Domino game has less graphic, so it needs basic operating computer.

As, it is implemented in java, it is platform free and should be possible to play it using any desktop operating systems.

The computer should have at least 512 MB memory and a single processor with more than 2 GHz clock to perform it faster.

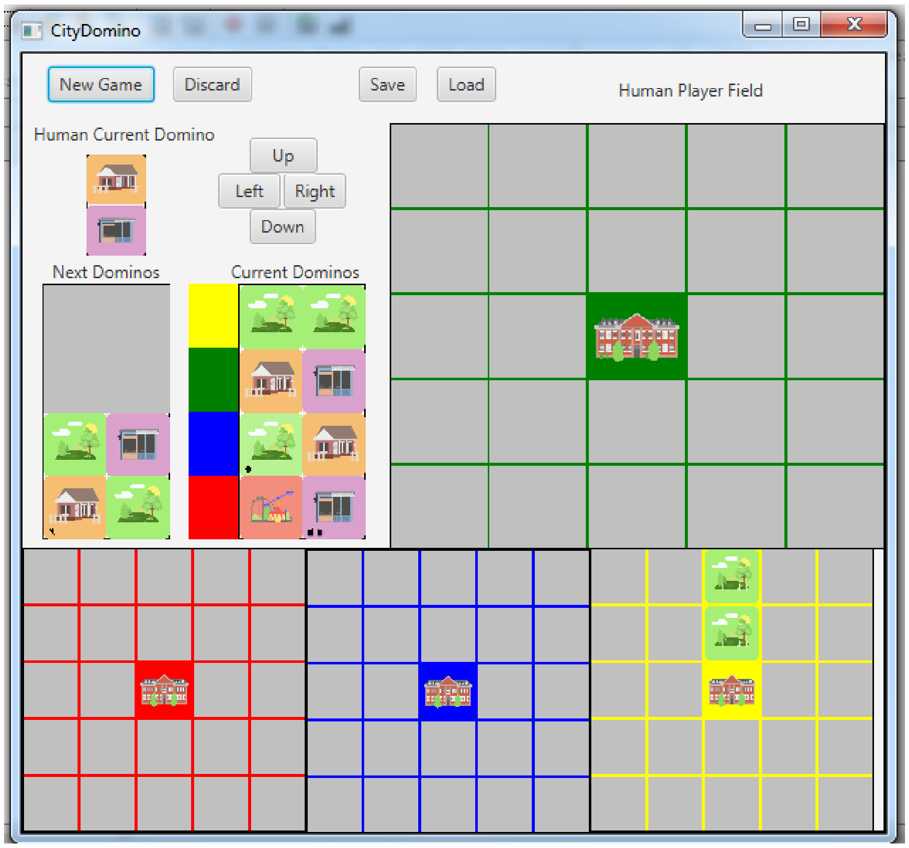
Dedicated graphic card/memory is not necessary, as it has less graphical features.

The game does not need any specific I/O devices to play it. It just needs ordinary mouse or keypad to play.

1-2 Program installation/start:

The game is provided in JAR file format and it does need to be installed on the computer. User just needs to store the JAR file into their fixed storage, opens it, and starts to play it.

1-3 Operating instructions:



GUI:

The GUI of City Domino game consists of several components which are shown above.

There are four **boards** with different background colors in the graphic. The bigger board with green background belongs to human player and it is mentioned on the top of it. The smaller boards belong to Bot players. The red one belongs to Bot1, blue one belongs to Bot2 and yellow one belongs to Bot3.

Next Dominos box is showing available dominos for next round that players should choose one of them for their next round. **Current Dominos** box is showing players' current dominos that they can place them onto their boards. Behind each domino in Current Domino box, the colors are showing which domino belongs to which player.

Up, Right, Down, and Left buttons are being used to move the cells to corresponding direction. The logic will check if the related row or column is empty to move the cells or not. If it is possible cells will be shifted to pressed button direction.

New Game button is being used to start a new game. It clears all the boards of the players, clears Current Dominos box, fills the stack of dominos, place city center in middle of each board and sets new dominos into Next Dominos box.

Discard button is discarding current domino of the human player. If its current domino is not null, next player will play.

Save and **Load** buttons are being used to save the current game and load a saved game from text file. If the format of the file is not correct program will not be able to load the game.

City Domino Game:

The city domino game is a game which four players design their city. The goal of the game is to design as many districts as possible with prestige. Each player has his own 5*5 cell playing field and starts with his city center in the middle.

A game bag for all players contains 48 playing cards in the size of two cells, each of which displays one (possibly the same) district type on its two halves. The district types differ in image and background color. Each playing card has a defined value. On some districts one to three prestige symbols are additionally shown.

Four cards are drawn and displayed in the first selection area. The card with the lowest value is sorted at the top, the card with the highest value at the bottom. The first player marks the card he would like to take in the selection area, the other players make their selections one after the other and mark the desired card.

If all cards have been marked, four cards are drawn again and also sorted in the second selection area.

Alternatively, the cards are always marked in the second selection area and as soon as each player has marked a card they are moved to the first selection area, including the markers, from which the cards are taken to the discard area. Newly drawn cards then only appear in the second selection area and are marked there.

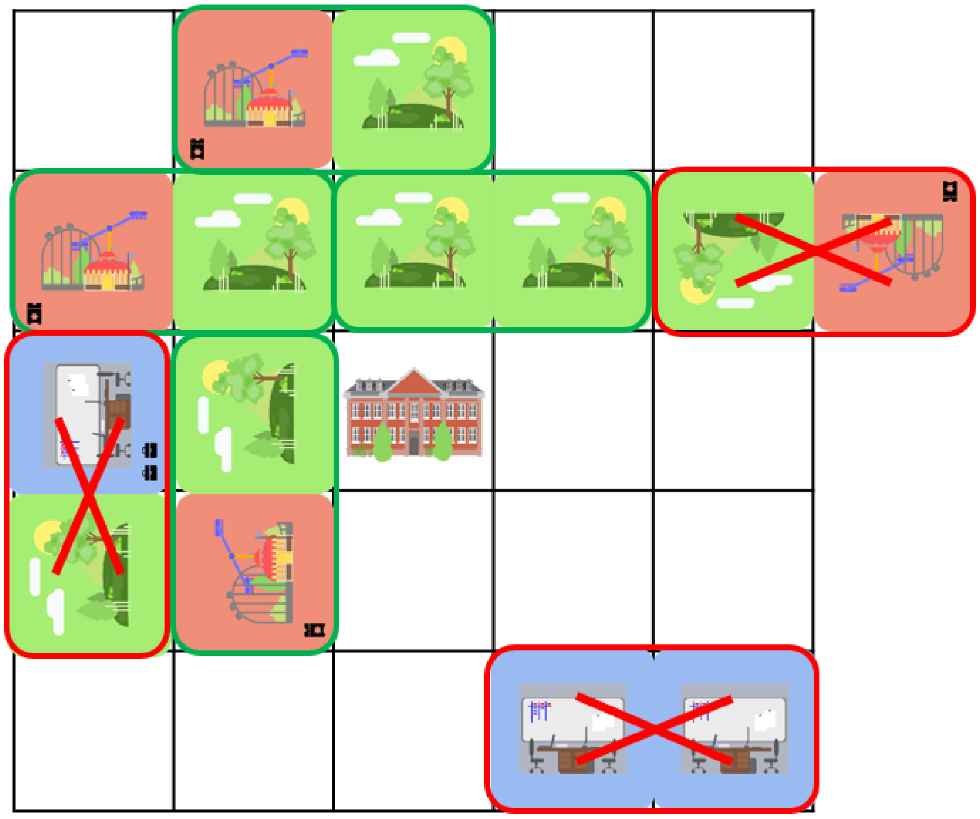
Game procedure
The player who has marked the top card in the first selection area begins a round, followed in turn by the player who has marked the card below. In a round, first a card from the second selection area is marked and is then selected for the next round. So the more valuable his marked card has in this round, the later the player's turn comes and the less choice he has for the next round. Then the player tries to place his card for the current round on the game field following the rules.

Attach Rules
The first card must border the city center. The city center may be bordered by any part of the city. Every following card must border either the city center or an other card on the field. If you place a card next to another card, at least one half with one side must border on an identical district type of the lying card.

If the card for the current round does not match either the city center or a card that is already on display, it is discarded.

All playing cards must fit into the 5*5 hex, no half may protrude. However, the city center does not have to be in the middle, but can be moved during the game, which means that all cards that have already been placed are also moved.

Once a card is placed on the field it cannot be moved individually, only together with the city center.

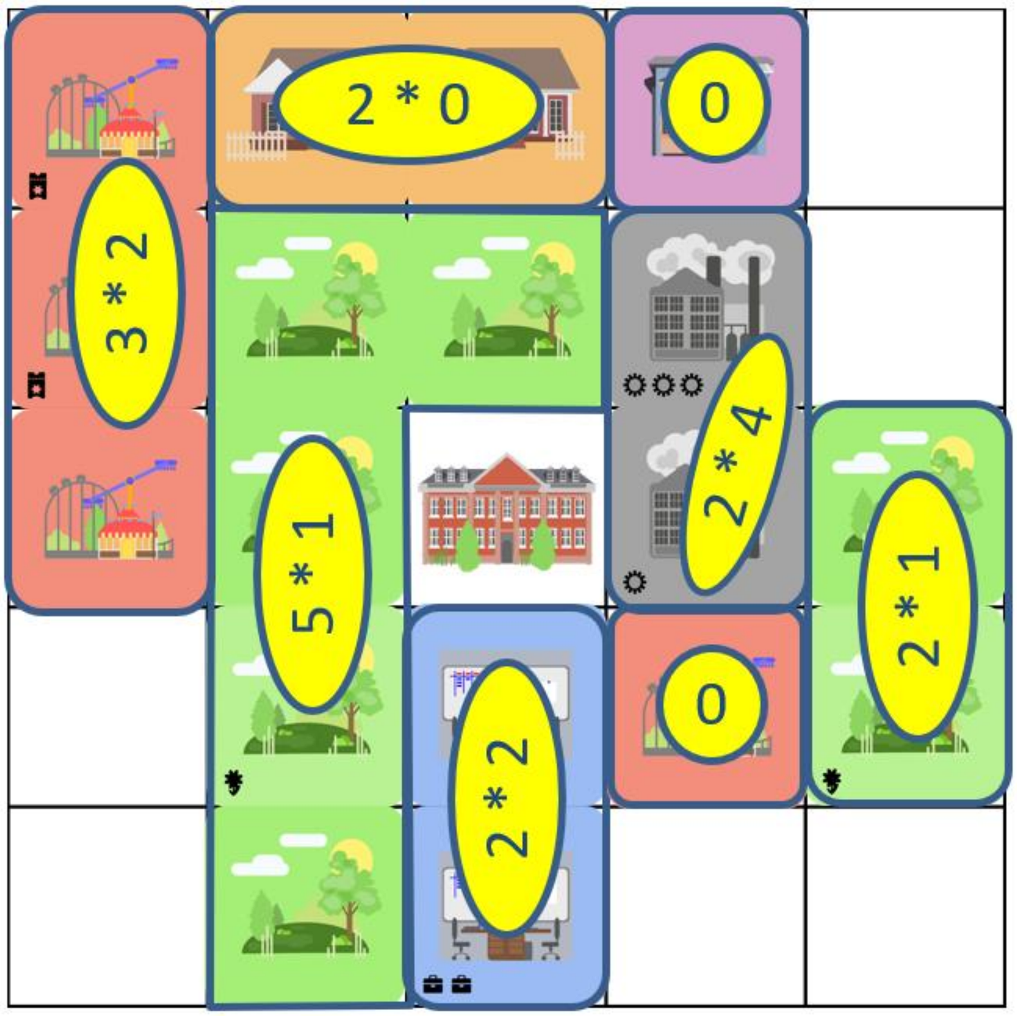


End of game

If all playing cards have been drawn out of the bag and placed or discarded by the selection areas on the playing fields, the points are determined.

- Each city consists of several districts. A district consists of horizontally and/or vertically connected cells of the same district type. The city center does not belong to any district.
- The points of a district result from the number of its cells multiplied by the number of prestige symbols it contains.
- Within a city there can be several separate districts of the same type. Each district must be evaluated individually.
- Districts without prestige symbols do not score points.

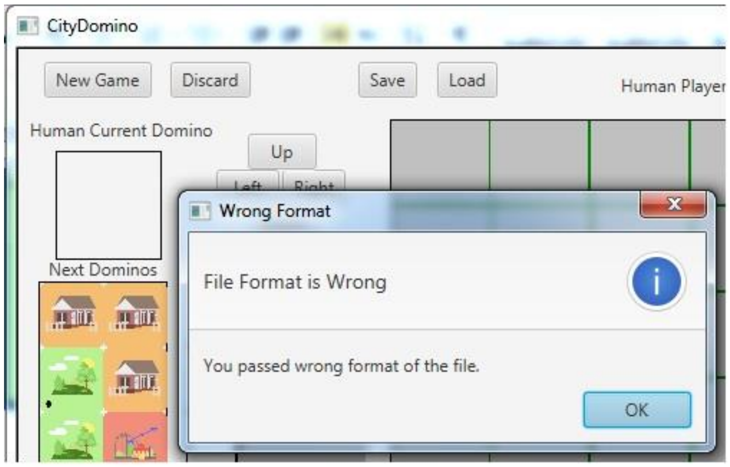
For the evaluation, the sum of the points for each player's areas is determined. The player with the most points wins. If there is a tie, the player with the largest single area wins. If there is also a tie, both players win equally.



1-4 here are all error messages:

The most common error messages are being happened during saving and loading. Especially when loading a file into the program the format of the file should be correct.

If the program finds more than 4 game fields in file, the file will not be loading into the game.

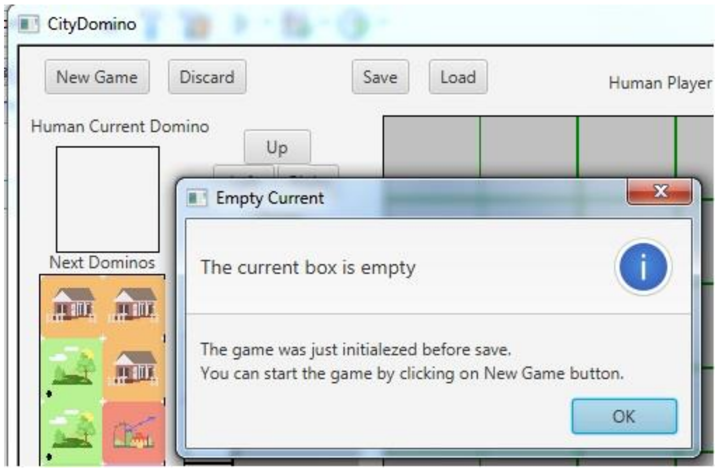


Wrong Format

File Format is Wrong

You passed wrong format of the file.

If there is also no current box and its place is empty the game will not be loaded and it will inform that he/she can start new game as not current box means game was save at the beginning of new game.



Empty Current

The current box is empty

The game was just initialezed before save.
You can start the game by clicking on New Game button.

2 Developer manual

2-1 Development configuration:

Netbeans IDE version 8.2 was used for program development and also java JDK was used for compiling and implementation. The code writing was done laptop with windows computer installed on it.

Junit version 6 was used for test of the program.

The project was saved into repository to access it from each computer at home or computers at lab of university.

For drawing the diagram of the classes and their connections, Microsoft Visio is used. It has more option to draw than Microsoft word.

2-2 Problem analysis and realization:

For the next choose box, it is found more user friendly that the user clicks on each domino he/she wants to play for the next round. The other solution was radio box, but putting radio box was taking space in windows.

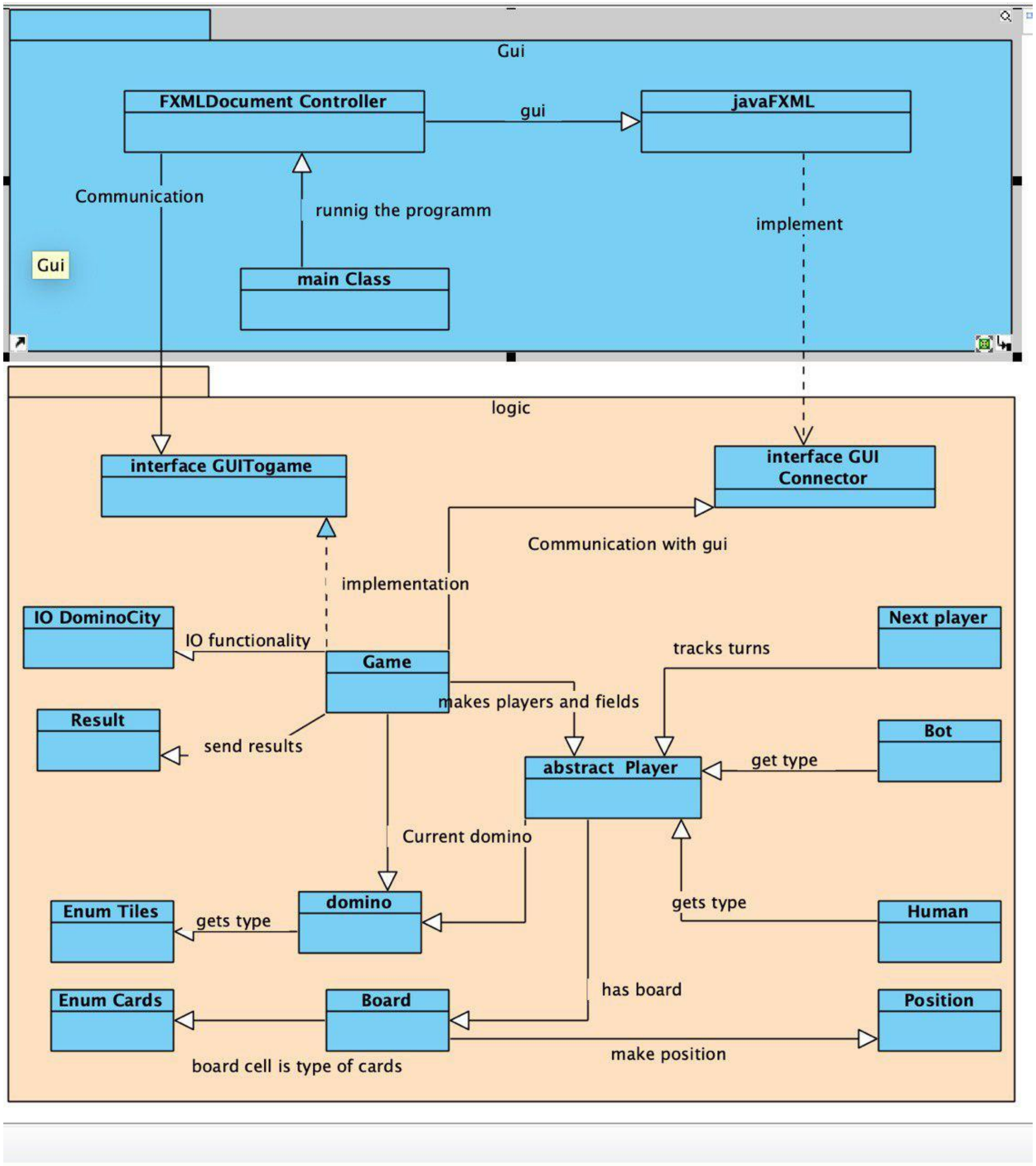
User can rotate its domino by clicking on domino itself. Also it was possible to use a button to rotate it.

For discarding the current card, there was to possibilities. First to have an to drag the domino into it or use a button. But in this program button is used as dragging is more difficult when user uses laptop.

Boards of the players are grid panes and inside them exists 5x5 image view. There was other option to put labels or buttons. But using grid pane is easier to get position for putting the domino. Also to put the domino drag and drop is used and it is better than clicking as by drag and drop user can simultaneously check if the domino is fitted into the board and then release it to place in its place.

Arrow keys are being used to shift the cells up, down, right, left. It allows user to move it only cell shift and prevent user to give bad input. It is also evaluated having arrow keys is better than moving cells by drag and drop.

2-3 Program organization plan:



2-4 Description of basic classes:

2-4-1 GUI

Game:

Game class in GUI package is the main starting point of the game. It creates the stage and loads the components which are made in FXMLDocument. Then it shows the graphics components to the user.

FXMLDocumentController:

In this class all the components of the gui were created and game will be initialized here and then it pass them to two objects of game in logic and JavaFXGUI. It also get users events and actions, and pass them to game in logic to update the logic and gui respectively.

JavaFXGUI:

This class is for updating any changes of the game in the logic package. An instance of JavaFXGUI is created in FXMLDocumentController and all necessary components are passed to JavaFXGUI constructor. When the changes happened in game of logic package, the updates are being sent to here via GUIConnector interface as this class implements abstract methods of GUIConnector interface.

2-4-2 Logic

Game:

This class is the main class in logic and an instance of this class in created in gui and all parameters to make a game is passed here. Players are being made in this class and their turn is being handled here. It receives users actions from gui and with coordination of other logic classes is controlling of the logics and pass the updates to gui to show it to user.

Board:

This class is creating a board for a player and implements all things that is necessary to work with board. It finds best position for given domino for bots, lay the given domino in given position, check if domino in given position make empty cell, check if the row are free to move the cards and move them, and calculate the point.

Domino:

This class is making domino based on the Tiles class in logic and save also the rotation of the domino. Based on the rotation, this class returns first and second tiles and also calculates the second position for the given first position of the domino. It also fills a stack which is a linked list with all tiles as domino.

Enum classes:

In package of logic there are three enum classes and they are being used to be places into the board or make dominos. The first enum is CardType and it stores all 8 types of the cards. The next enum is Cards and it stores cards which have to attributes, prestige (point) and type from CardType enum class. Then, we have Tiles and each tile contains to cards, totally 48 possible tiles to make 48 dominos to play.

Player, Human and Bot:

The class Player is an abstract class and classes of bot and human inherited this class to make human and bot players. Human class is inherited Player class and making an instance of human player. The Bot class is also inherited Player class and is making bot player. These three classes have some attributes like name, id, board, current domino and next domino and game class can call them for each player to implement accordingly.

Pos:

The class Pos is making positions with given x and y coordinates of the board. It also calculated neighbors of given position and also checks if two positions are next to each other. Positions are used in the game to put cards into boards of players.

Interfaces (GUIConnector and GUItoGame):

In logic, there are two interfaces, GUIConnector and GUItoGame interfaces. GUIConnector interface has methods which are necessary to communicate from logic to gui. After an update make in gui respective method should be called to update the gui based on the changes. JavaFXGUI class is implementing GUIConnector interface class.

GUItoGame interface has methods which are needed by gui to pass users actions and events on graphic to logic. Then in logic appropriate update will be done and changes will be passed again to gui to update graphical board.

IOCityDomino:

This class is being used when user decides to save the current game into a text file or load a game from stored text file. For loading, it first stores all given information of the text file in string format and then create instances of the logic classes to change string to this instances. Finally, it creates a new game and pass all the information to a constructor to construct a stored game.

Result:

When game is finished, an instance of result class is created and the array of players is passed to here to find out who is the winner of the game, and its score. Then, this information can be used in gui to show it in a pop-up windows.

2-5 Program testing:

test	result
Arrow key when banks are free, cells should move.	Works as expected. When the player tries to shift the cells, program check if the related bank is empty and cells will be shifted
Arrow key when banks are not free, cells should not move.	Works as expected, when banks are not free, cells will not move.
Domino not touch city center at beginning	Works as expected. Players can not put their first card without touching city center.
Domino parts not touch same type	Works as expected. Player cannot put their dominos next to dominos if neither parts are the same.
Human player chose domino, can’t choose again	Works as expected. When human has already chosen a domino, he/she can not choose again.

Human player can’t choose empty dominos of next box	Works as expected. When dominos are chosen by player their placed are empty.
Human click discard, next players should play	Works as expected. When human has its current domino in selection area, he can not to play it by clicking on discard/delete button.
Human can rotate his/her domino	Works as expected. When human click on his/ her current domino. The domino is turned
Player cannot load file with less or more than 4 field	Works as expected. Player cannot load a file that has more than 4 fields or less than 4 fields.
Player cannot load a file that was generated right after a starting game.	Works as expected. When player try to save a game that was just created and load again, he/ she will receive a message that they can start new game instead.
User cannot drag into Bot fields	Works as expected. User can only drag its domino into his/her own field and cannot put it into Bots fields.
Winner with points will be shown at the end	Works as expected. At the end of the game, the winner name and its point will be shown in a pop up window.