CS 319 - Object-Oriented Software Engineering

Final Report

Battle City

Group 1-B

Kaan Sancak

Mahin Khankishizade

Ozan Kerem Devamlı

Supervisor: Ugur Dogrusoz

İçindekiler

1. Implementation Process 3

1.1. System Requirements 4

1.2. User's Install Guide 4

2. Changes & Improvements 5

2.1. Design Changes & Improvements 5

2.1.1. Enemy Improvements 6

2.1.2. Bonus Improvements 7

2.1.3. Portal Improvement 7

2.1.4. Statue Improvement 8

2.1.5. Multiplayer Improvement 9

3. User's Manual 10

4. What is Missing? 13

5. Conclusion 14

1. Implementation Process

The implementation process started right after the first iteration finished. We decided to implement our project with the help of IntelliJ. Each of us downloaded IntelliJ and connected to our main GitHub depository named 1B.battlecity-cs319, which was created by Kaan in the beginning of the course. The main advantage of this program is, it is directly connected to GitHub and has an access to the classes written or created in the project. Each time when one of us fixed the class, he/she could easily push it to GitHub and merge it with others' codes if necessary. In addition to GitHub we also used the social media programs as Skype and Discord to communicate with each other when we are not in campus.

We did not divide the tasks among each other at first, rather we decided that each of us will try their best in the area that they are familiar with. Thus, at first, Kaan implemented basic structures of all classes. Later, Ozan and Mahin added methods to them. After this point, we decided to divide the classes since this method seemed more efficient to us. Kaan was working on the logic of the game, whereas Mahin was working on the user interface and Ozan helped Kaan with the game logic. When the second iteration finished, we were already done with the design of the game and the most part of the logic. Thus, we added more functional requirements and began to implement them. As a result, we exceeded our expectations from the game, and could finish the project in time.

In the beginning of the second iteration, we noticed that our implementation is not corresponding to the design that we mentioned in the report. The objective of

this course was to implement the code according to the analysis and design reports, so we decided to change our code and try to implement it according to what we have written. Thus, our GameManager and MapManager classes were implemented as according to 'Singleton' design pattern. Then, we added 'Player-Role' design pattern to between Tile and Portal classes and also among Bot classes. We also tried to stick to Object Oriented Programming as much as possible.

In general perspective, we have finished all of our promised functional requirements and we also added some additional functional requirements like Statue, extra bonus types and extra bot enemy types. Rest of the report will mostly mention these and also some guides to install the game.

1.1. System Requirements

Battlecity game will require JVM(Java Virtual Machine) and Java SDK. Any computer which has these will be able to install and run the game.

1.2. User's Install Guide

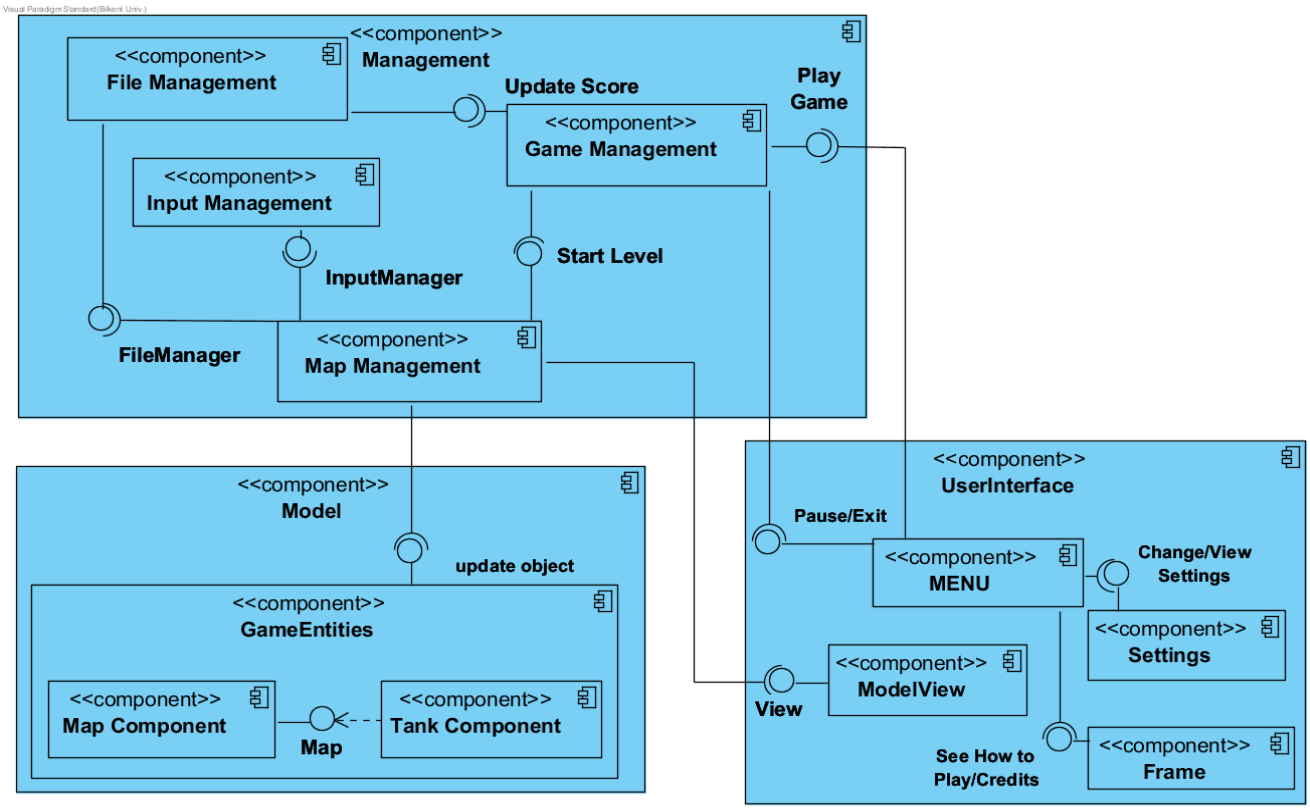
- Enter to the link below to see our reports and project
  https://github.com/kaansancak/1B.battlecity-cs319
- Clone the project via command line or bash or Download the project as a ZIP file and run it in any Java Compiler (for example: IntelliJ, Eclipse etc.)
- To learn how to play game users can check "How to Play" option the menu.
- To see credits, players can "Credits" to learn who developed the game.
- For learning more about our implementation, check our second iteration reports

2. Changes & Improvements

2.1. Design Changes & Improvements

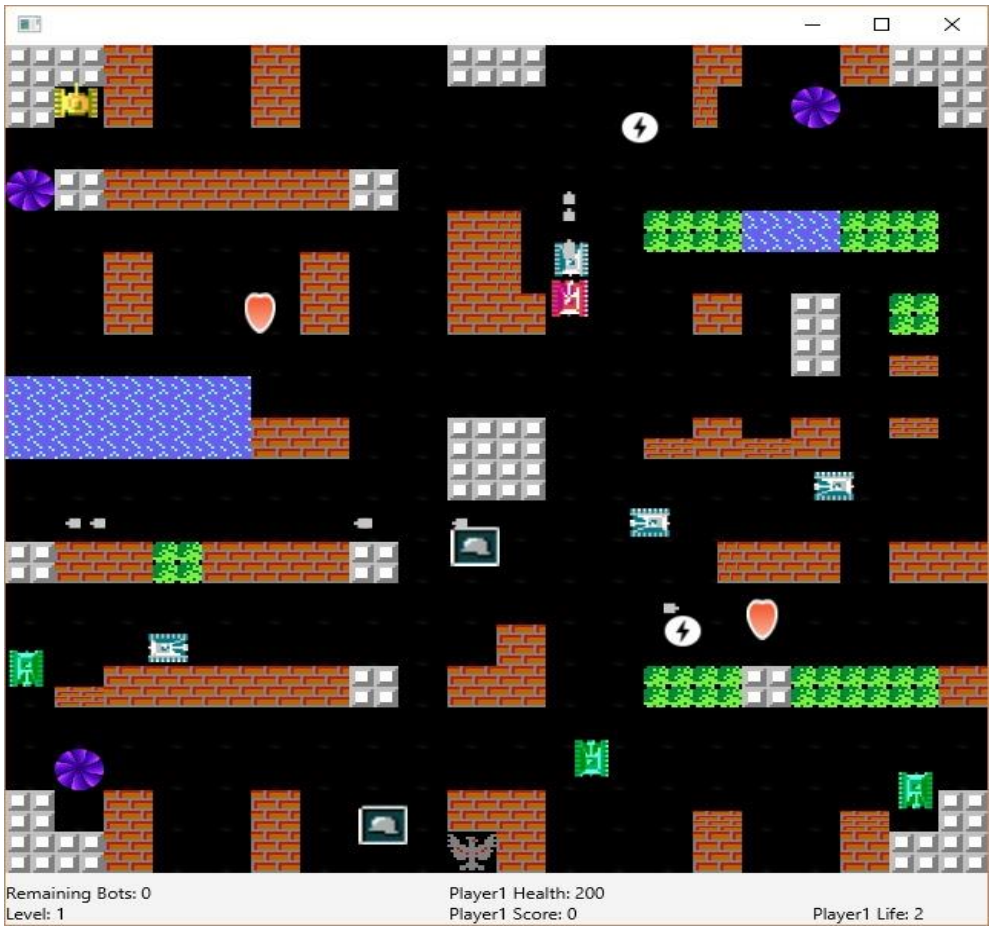
If you look the first iteration of our design report you can see that our classes were not properly ordered. That's why, in the second iteration we have re-do our system decomposition and come up with more efficient system decomposition whereas our main architectural pattern stayed as MVC. Since our decomposition is changed we re-arranged our packages according to our design in the second iteration of the design report. During implementation, we tried to stick to our design as much as possible.

Most importantly, as mentioned before we applied "Singleton Design Pattern" for Game Manager and Map Manager classes since we wanted only one handler/controller for active game and active map. We also additionaly added GameStatus enumeration class to check the current status of the game among manager classes. Our design is improved as follows:



2.1.1. Enemy Improvements

Normally, we have promised just one type of enemy with basic AI. However, since we finished our promised functional requirements before deadline. We have added 2 more enemy types with different images, health and armors. Since, from the beginning we applied the principles of object oriented design, it was very easy to implement these additional enemies. Most importantly, we used **Player-Role** pattern among enemy Bot classes. You can see different types of enemies from figure below.



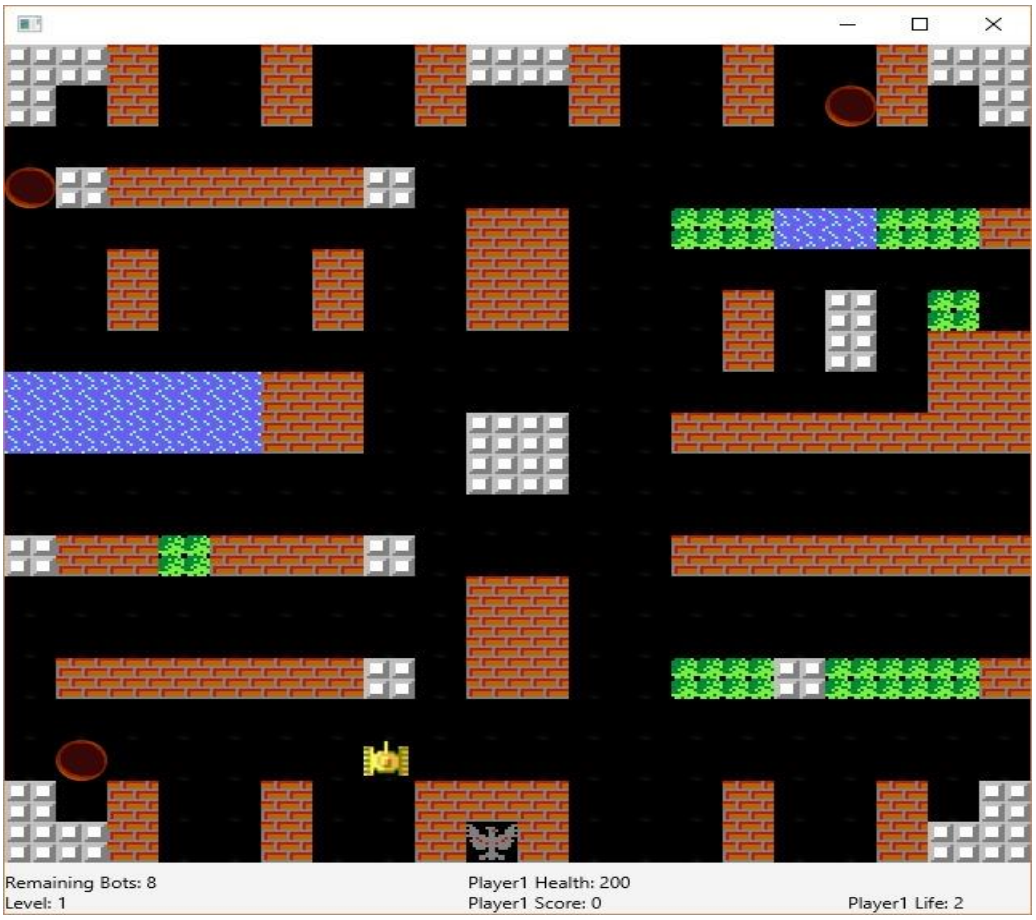
2.1.2. Bonus Improvements

Normally, we have promised 2 types of Bonuses in the second iteration of the design report which are life and speed bonuses. Additionally, we have added armor bonus which gives the additional life to player who takes the bonus. From figure above you can see all three types of bonuses.

2.1.3. Portal Improvement

We have added a portal tile. When player comes to a portal, portal teleports player to another portal in the game, randomly. From the figure above you can see the

active status of the portals. Portal also can be deactivate, in this status portal do not teleport players. Here is an image of deactivate portals.

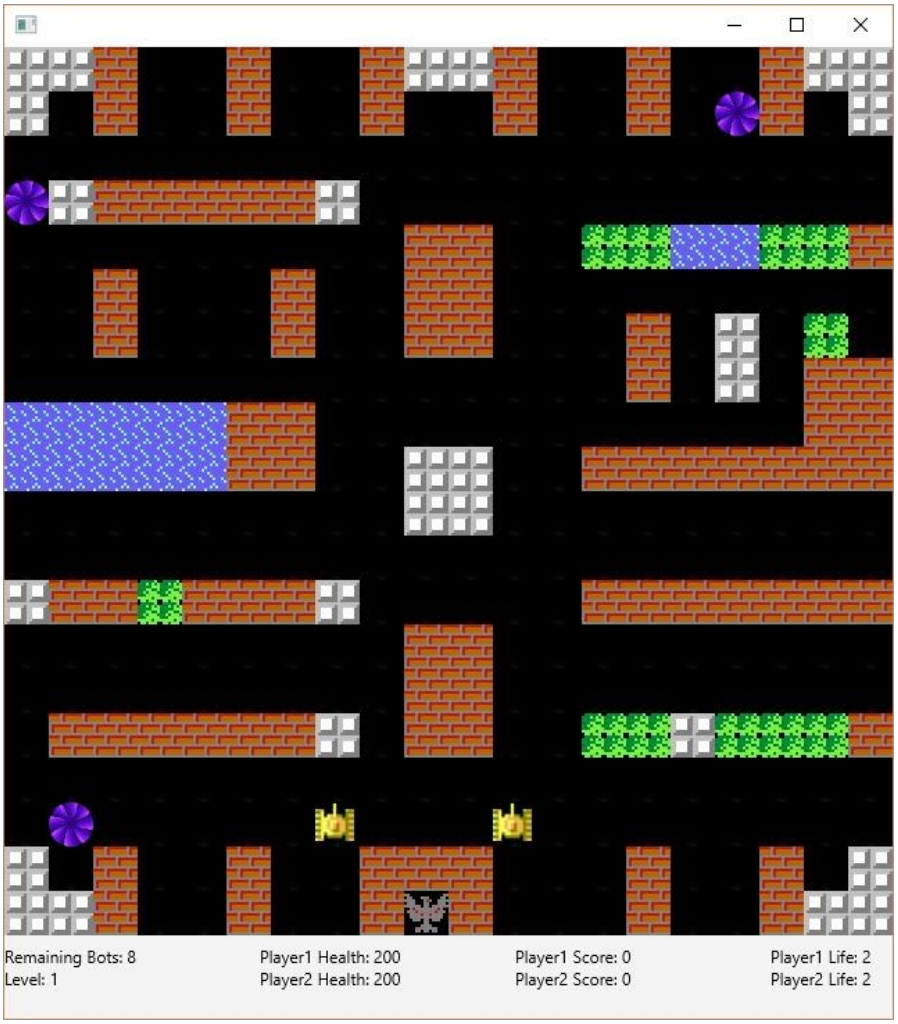


2.1.4. Statue Improvement

We have added a statue object to the model as in the real game. When the statue gets destroyed by bots, player loses the game. You can see the statue from the figure above(Eagle Figure).

2.1.5. Multiplayer Improvement

In the first iteration, we could not complete the multiplayer game option. We have managed to finish it in the second iteration. You can see the multiple player options in the menu view.



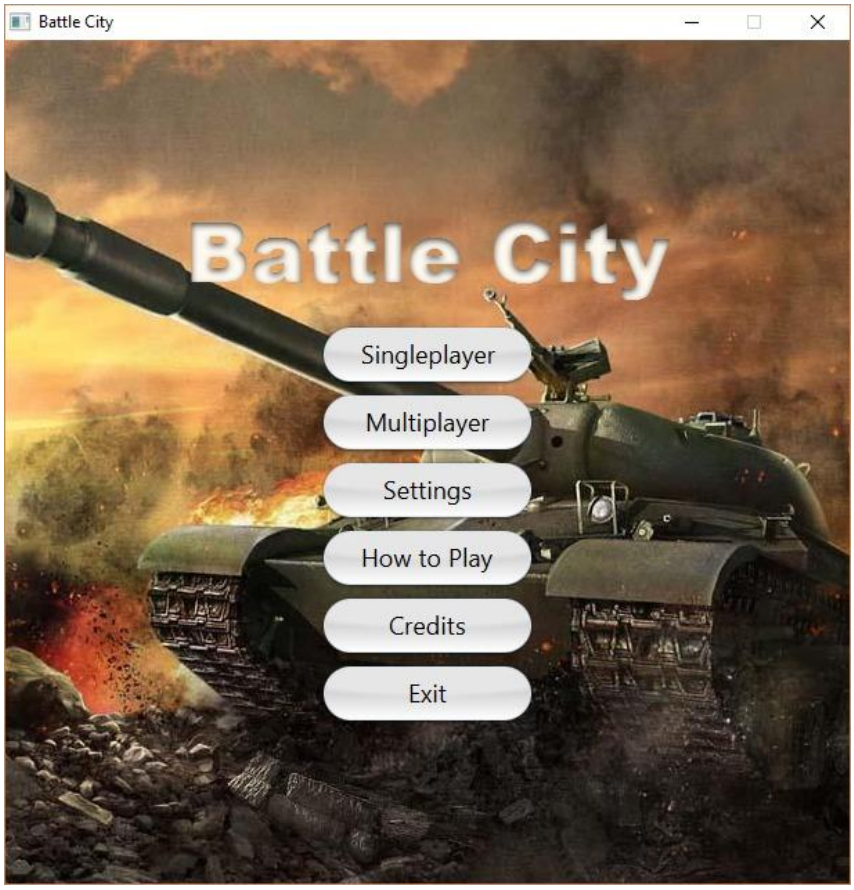
3. User's Manual

Installation manual is given sections above, this is a in game manual.

a. How to Start

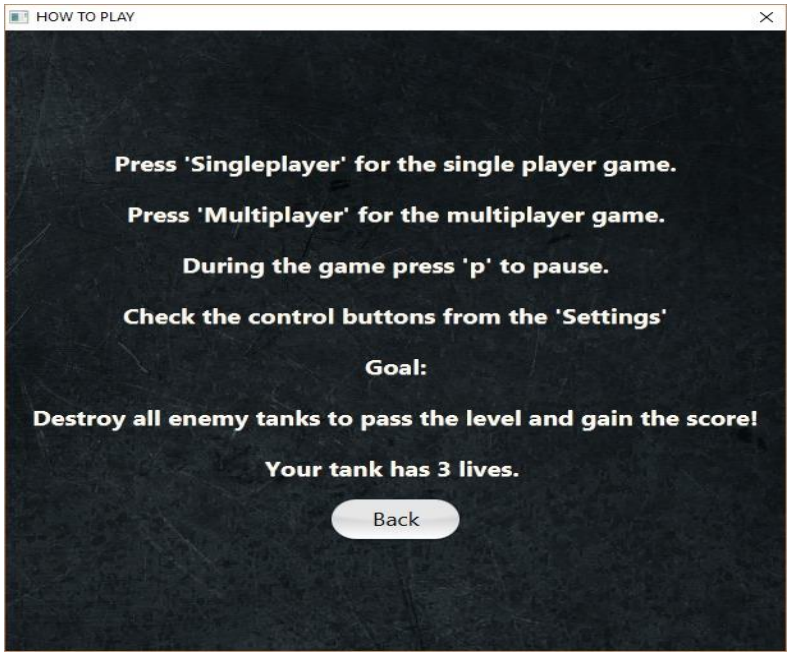
For briefty, menu screenshot does not given since it given above.

Users can different selections from the main Menu. It is the first screen of the game. Singleplayer option starts a new game with single player whereas multiplayer game starts a game with two players.



b. How to Play

Players can learn how to play the game from How to Play Screen, detailed information of the player keys are given in “How to Play”.

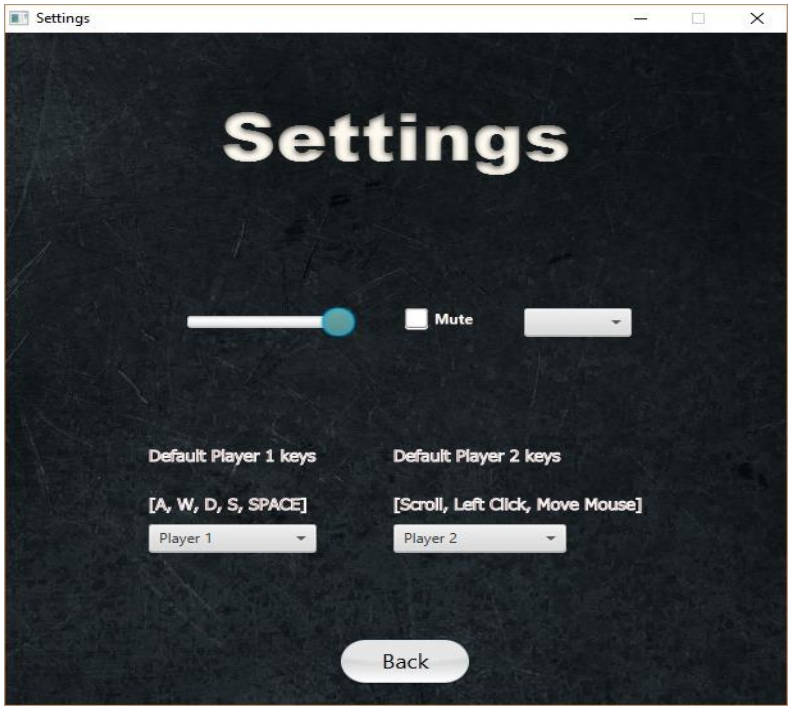


c. How to Proceed

Players can proceed in the game by destroying all remaining bots in the game. Remaining bot count is given in the info label during the game. When a level is finished users are asked to continue. If players die before destroying remaining bots or if bots destroy statue the game ends.

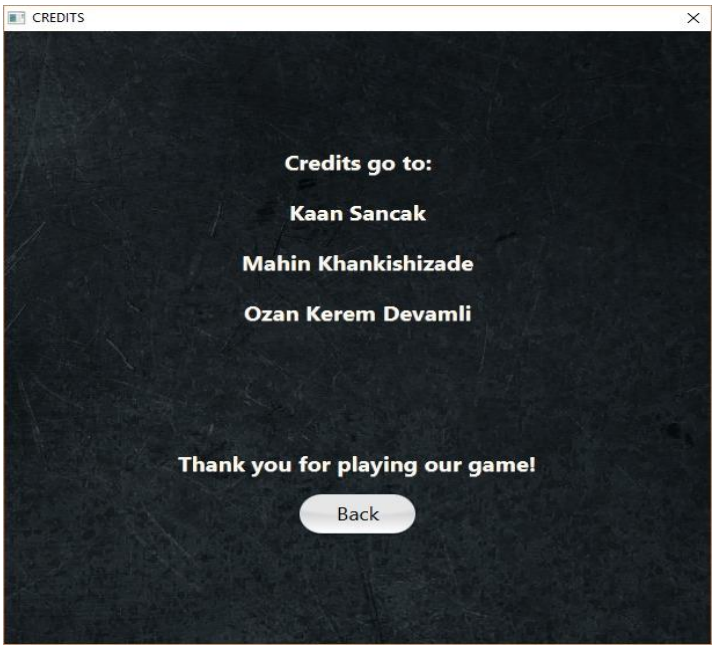
d. Settings

User can change/see their settings from settings.



e. Credits

User can see the credits of the game from credits.



4. What is Missing?

We have finished our promised functional requirements so there is not much missing but during implementation we might have skip some design goals. We have also added additional requirements as mentioned.

5. Conclusion

In general perspective, we think we did a great job. We tried to stick to "Principles of Object Oriented Design" as much possible. Doing the analysis and design first, was very different for us since we usually do just implementation. During these processes we have observed the advantages of analysis and design. Generally, we saw that when we have the design, implementation is nothing but coding, meaning it is simple. However, we also see that working with a group has both advantages and disadvantages. While splitting the work load is very good, it is very hard to proceed when one or more member fails to do their duty.

As a conclusion, we have tried our best to learn from this project and complete our tasks accordingly.