**HERZIG CHESS PROJECT**

Developer: Shaaban, Mohammed Najib Ahmed

Contents

[**Part I: Analysis and Design** 3](#_Toc433191)

[1.1 Minimum Hardware Requirements: 3](#_Toc433192)

[1.2 Minimum Software Requirements: 3](#_Toc433193)

[1.3 Instructions: 3](#_Toc433194)

1.4 Functional Requirement: ……………………………………………………………………………………………………4

[1.5 Game Rules 9](#_Toc433195)

[**2.** **UML DIAGRAM** 12](#_Toc433196)

[2.1 Use Case Diagram 12](#_Toc433197)

[2.2 Class Diagram 13](#_Toc433198)

[2.3 Sequence Diagram 14](#_Toc433199)

[**PART II: Java Implementation & Documentation** 15](#_Toc433200)

[**3.** **Programming Information** 15](#_Toc433201)

[**4.** **Classes** 18](#_Toc433202)

[**5.** **Design Pattern** 18](#_Toc433203)

**Part I: Analysis and Design**

## **Minimum Hardware Requirements:**

* Monitor
* Keyboard
* Personal Computer (PC) or Laptop
* Mouse

## **Minimum Software Requirements:**

* Windows OS
* Mac OS
* Notepad++
* Software to unzip file, eg:Winrar, 7zip

ss

## **Instructions:**

Extract the ZIP file to any location in the PC.

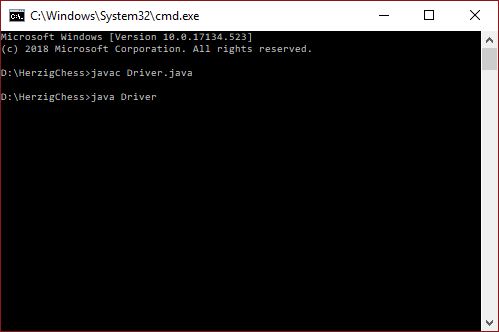
Run command prompt from the exact location where the file was extracted.

Type in “javac Driver.java” to compile, followed by “java Driver” to run the program.

Example: In our system. The zip file is extracted in the Data (D:\HerzigChess)

Then run the command prompt.

The instruction of command prompt is as shown below



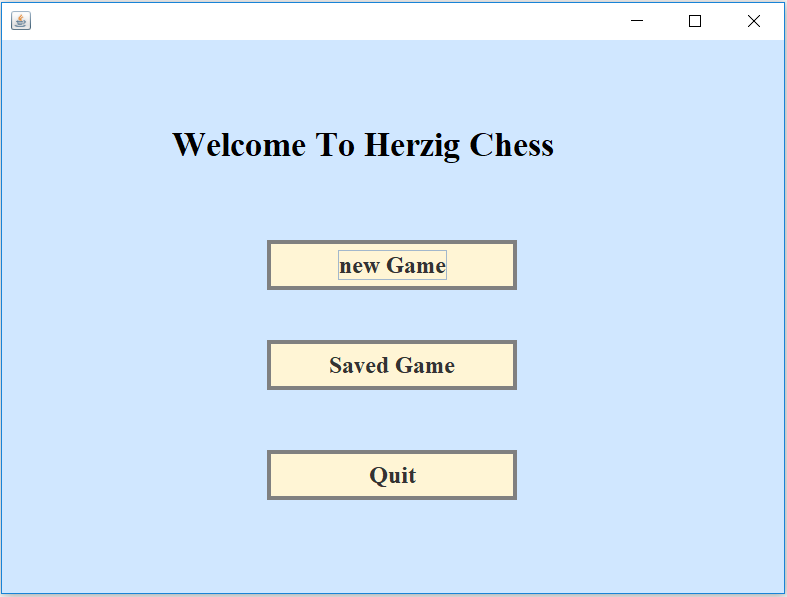
*Figure 1.0 shows the command prompt*

* 1. **Functional Requirement**

1. **Initialize game**
   1. **Start new game**

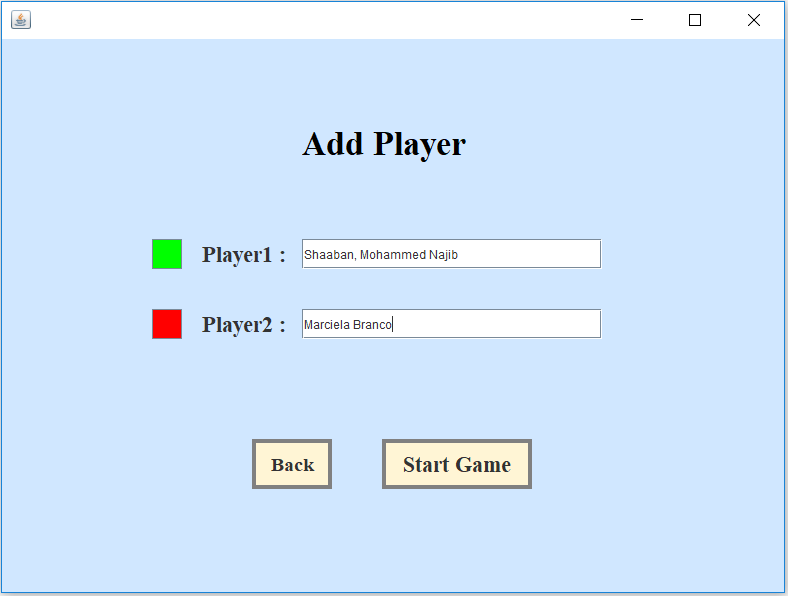
User have the option to start new game any time the program is run instead of loading pre saved game.

* Press the 'new Game’ button to start a new game.
* Press ‘Saved Game’ button is to load the list of pre saved game.
* Press ‘Quit’ to exit the game

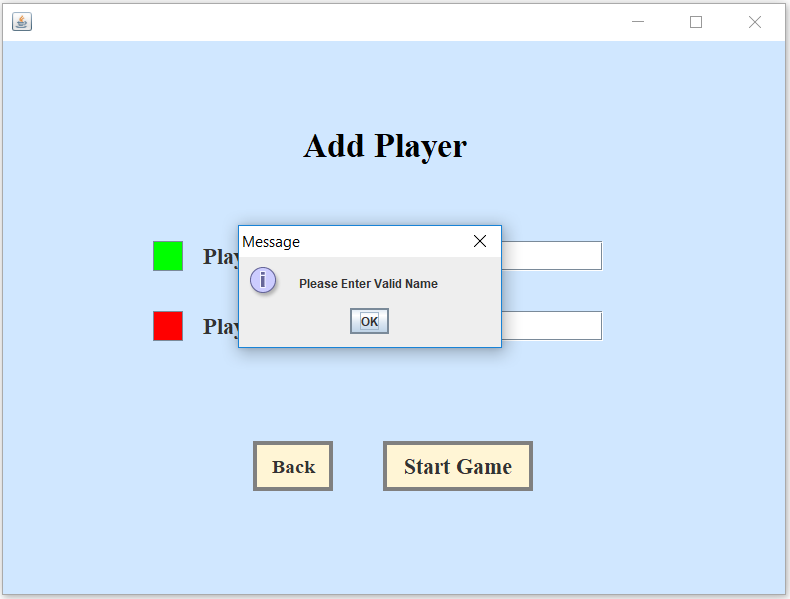


* 1. **Add players**

The player should input their names in the game menu screen. The system should save the credentials of the players. The system should indicate the assigned chess piece color for each player.



The Game Should display an Error if the user try to start game without valid name

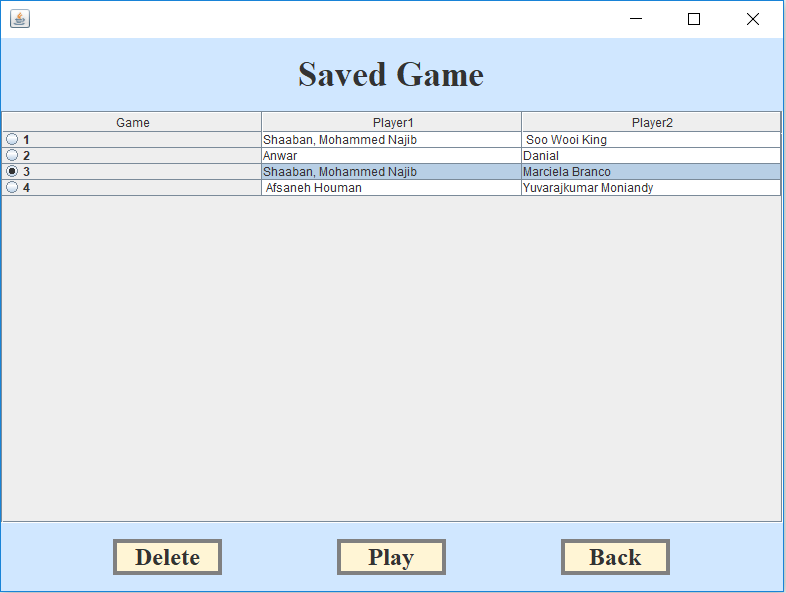


* 1. **Load pre-saved game**

The system should read through its database to identify a pre-saved game. When there exists a pre-saved game, the system should provide options to the user whether to create a new game or continue with the pre-saved game. When the user indicates to load a pre-saved game, the system should read the database and initialize the saved game state into the game screen.

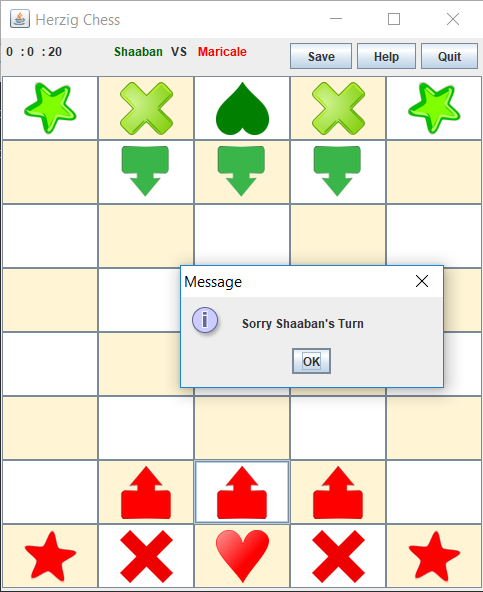
The interface below is to display the list of pre-saved game.

* Press ‘Delete’ button to delete the pre-saved game
* Press ‘Play’ button to loed and start continue playing the pre-saved game.
* Press ‘Back’ button to go back to the intila screen.
* Players has to indicate which game is to be played or deleted



* 1. **Declare which player’s turn**

When the player indicates to start the game, the system should load the game screen. The system should indicate which player’s turn it is to make the move. When it is not a player's turn, the system should disallow the player from making a move



* 1. **Record time**

The system should display the elapsed time from the beginning of the game.

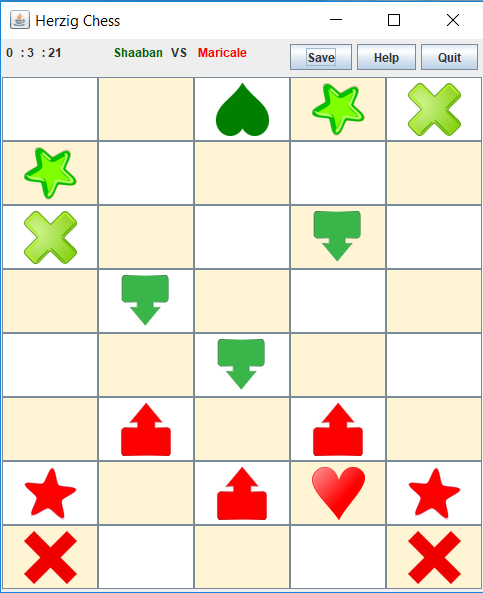
1. **Play game**

**2.1 Select and unselect piece**

The player during his turn should a select valid piece to move. The system should disallow the player from selecting an invalid piece such as the opponent’s piece. When the player selects a valid piece, the system should indicate the selected piece. When the player requests to unselect a piece, the system should unselect the piece.

**2.2 Move Piece**

When the player selects a valid destination box, the system should remove the piece from its initial position and place it on the destination box. The system should toggle the turn to the other player.



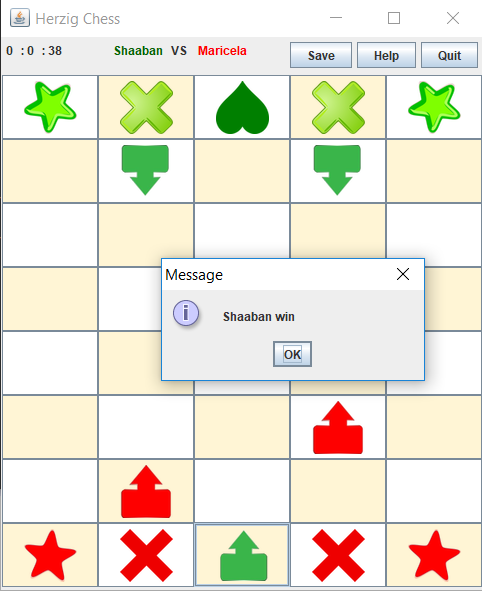
**2.3 Transform piece**

The system should transform each pieces into a different pieces as shown in the rule after 3 turns.

**3) Terminate game**

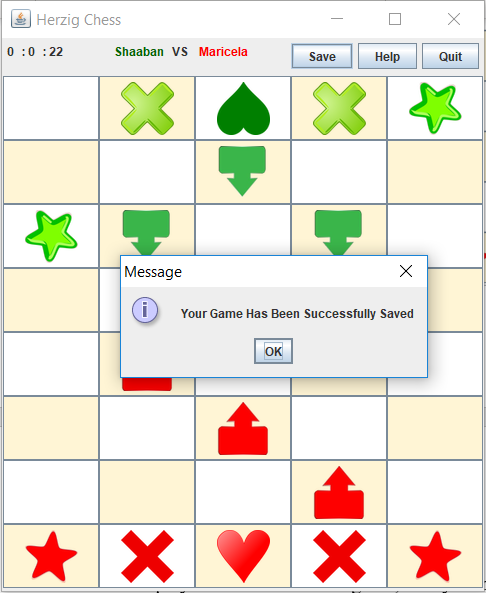
**3.1 Terminate the game and declare the winner**

When the sun piece of one player is captured by another player, the system should terminate the game and declare the winner.



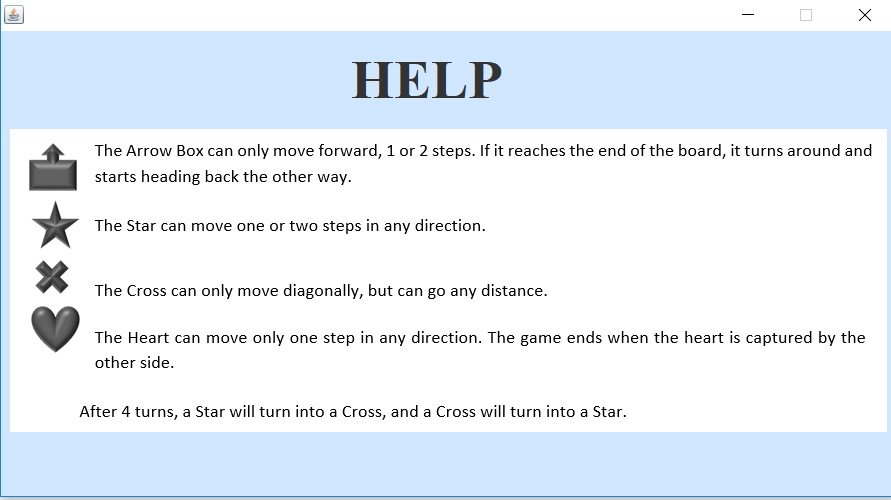
**3.2 Save the game**

the player chose to save the game, the system should save the current state of the game (pieces position, time, etc) and display a message to inform the user that the game has been saved secssfully.



**4. Display Help**

The Help button enabe the user to display information about game rule and how to play.



## **1.5 Game Rules**

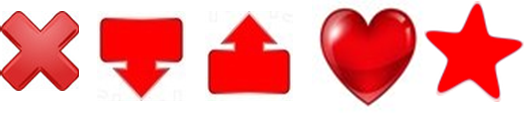
The Herzig Chess is a multiplayer Chess game. The green player as player 1 and the red player as player 2. The green player will start first then follow by the red player.

**Player 1 Chess pieces**



*Figure 1.5 shows the chess icons for player 1*

**Player 2 Chess pieces**



*Figure 1.6 shows the chess icons for player 2*

* The Arrow Box can only move forward, 1 or 2 steps. If it reaches the end of the board, it turns around and starts heading back the other way.
* The Star can move one or two steps in any direction.
* The Cross can only move diagonally, but can go any distance.
* The Heart can move only one step in any direction. The game ends when the heart is captured by the other side.

The Chess pieces will transform when the piece is being played. After 4 turns a Star will turn into a Cross, and a Cross will turn into a Star.



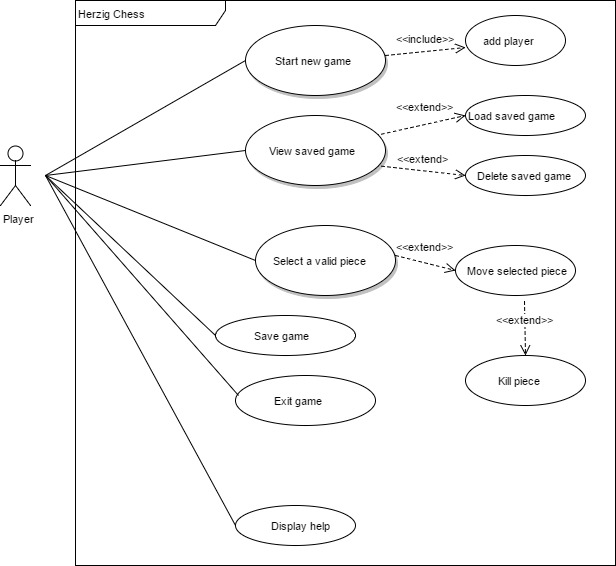
*Figure 1.7 shows a Start icon turns into a Cross icon.*



*Figure 1.8 shows a Cross icon turns into a Star icon.*

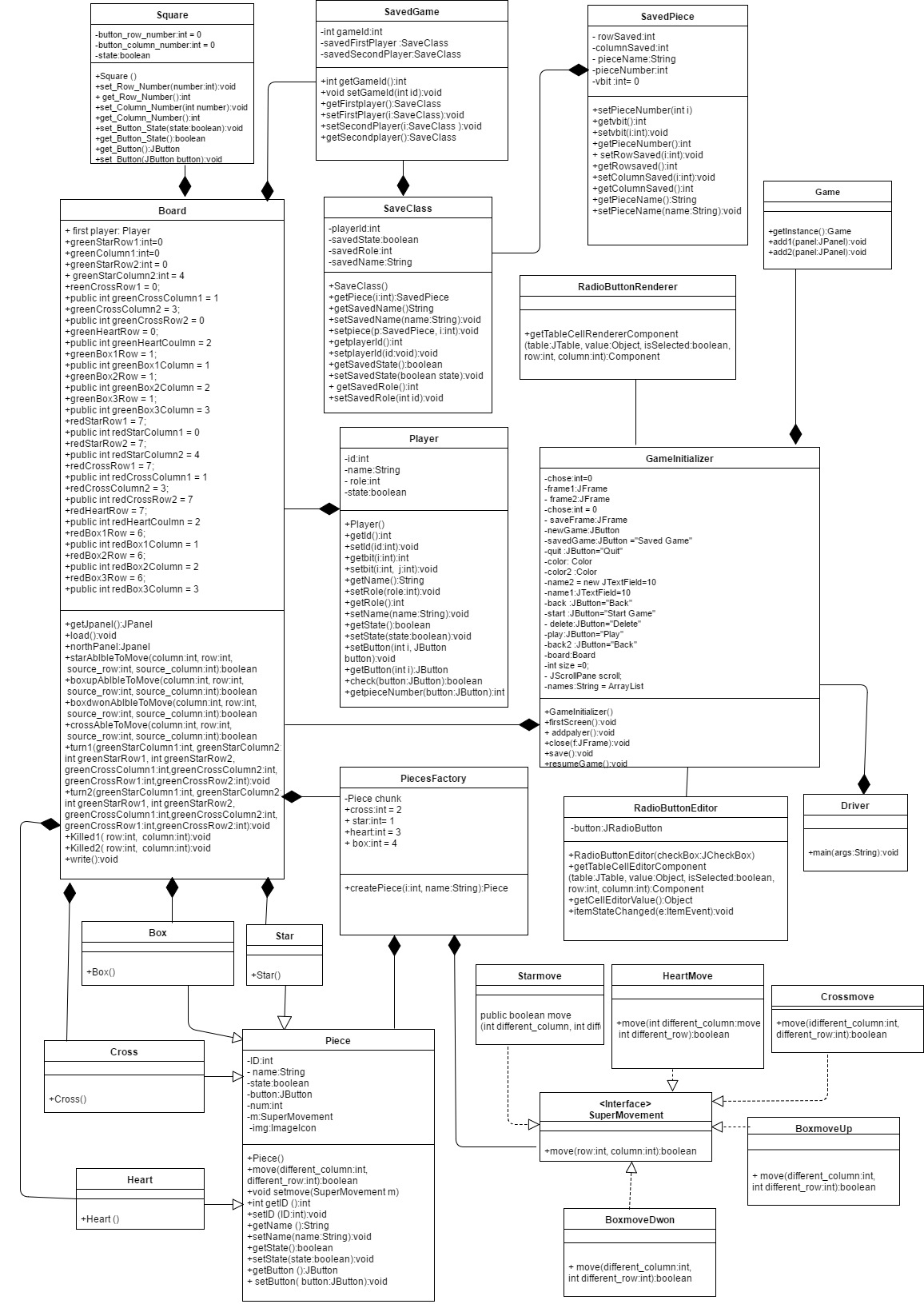
# **UML DIAGRAM**

## **Use Case Diagram**



*Figure 2.0 shows the use case diagram*

## **Class Diagram**

**

*Figure 2.1 shows the class diagram*

## **C:\Users\User\Documents\Degree subject\HerzigChess\Report\sequence.pngSequence Diagram**

*Figure 2.2 shows the sequence diagram*

# **PART II: Java Implementation & Documentation**

# **Programming Information**

* 1. The GUI of the main Menu will be displayed when the program is running.

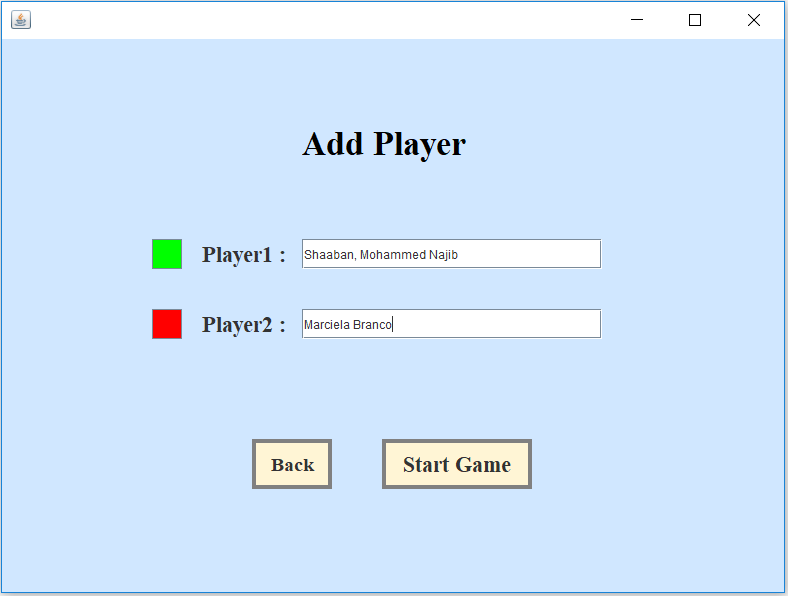
The Herzig Chess interface will display the “New Game” button, “Saved Game” button and “Quit” button.



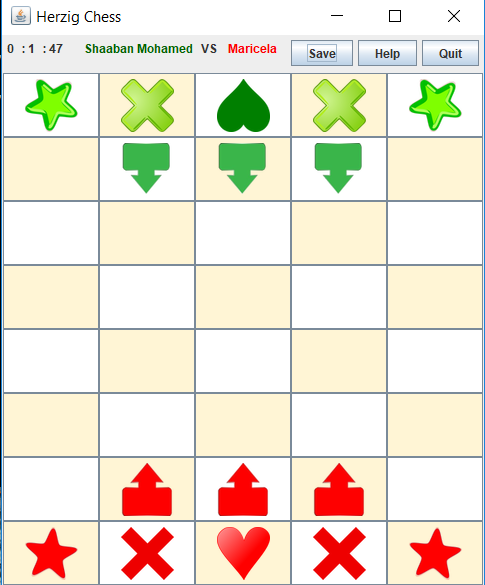
* 1. When the “new Game” button is pressed the program will display the add player interface.

The add player box will ask the players to add their name to start the game the game reject to start until players insert a valid name.

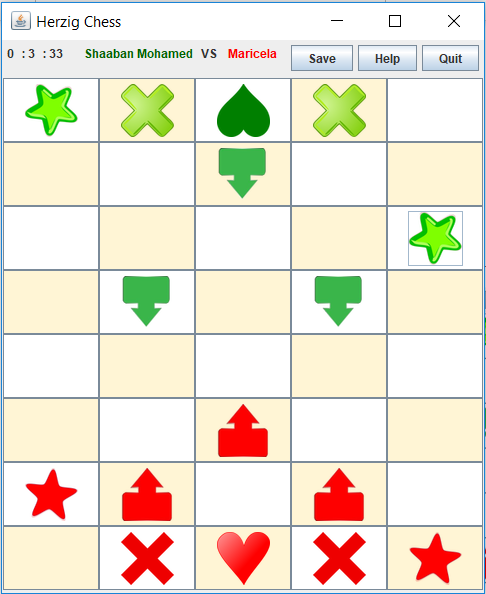
.



* 1. After the players add their name, the game interface will be display and the two players begin to play.



* 1. When the chess piece moves after four turns, the Star will turn into Cross and the Cross will turn into Star.

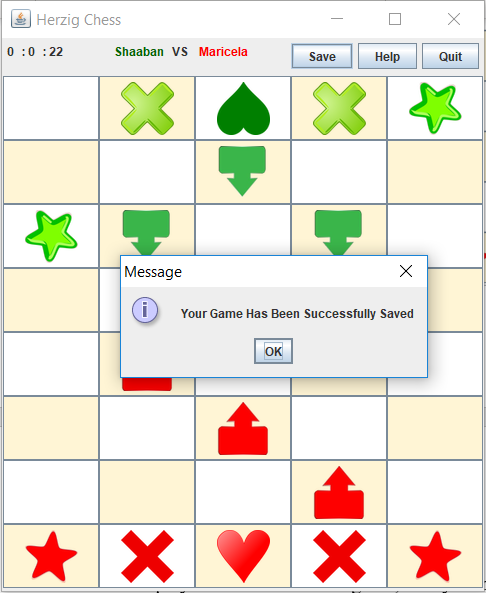


* 1. End Game scenario: When the Heart piece of either side is capture then it is a game over and the side that capture the Heart will declare as a winner.

As shown below, the Player 2 has capture the Heart of the Player 1, which means Player 2 win the game.



* 1. Saved Game: Players can choose whether to Save the Game after done playing or players can Save the Game during playing, which allow them to return to the game whenever they wanted.



# **Classes**

Notepad++ is used to develop the program. The program has 22 concrete classes, 1 interface class.

The concreter classes:

Board.java, Box.java, Boxmovedown.java, BoxmoveUp.java, Cross.java, Crossmove.java, Driver.java, Game.java, GameInitializer.java, Heart.java, Heartmove.java, Piece.java, PieceFactory.java, Player.java, RadioButtonEditor.java, RadioButtonRenderer.java, SaveClass.java, SaveGame.java, SavePiece.java, Square.java, Star.java, Starmove.java.

The Interface class:

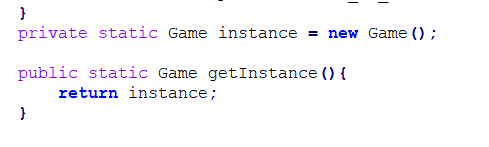
SuperMovement.java

1. **Design Patterns**

1. **Singleton**

The singleton pattern is a software design pattern that restricts the instantiation of a class to one. This is useful when exactly one object is needed to coordinate actions across the system. The term comes from the mathematical concept of a singleton

Singleton design pattern is applied in Game class to ensure only a single instance of the class is created. The following piece of code explain how the pattern is created.

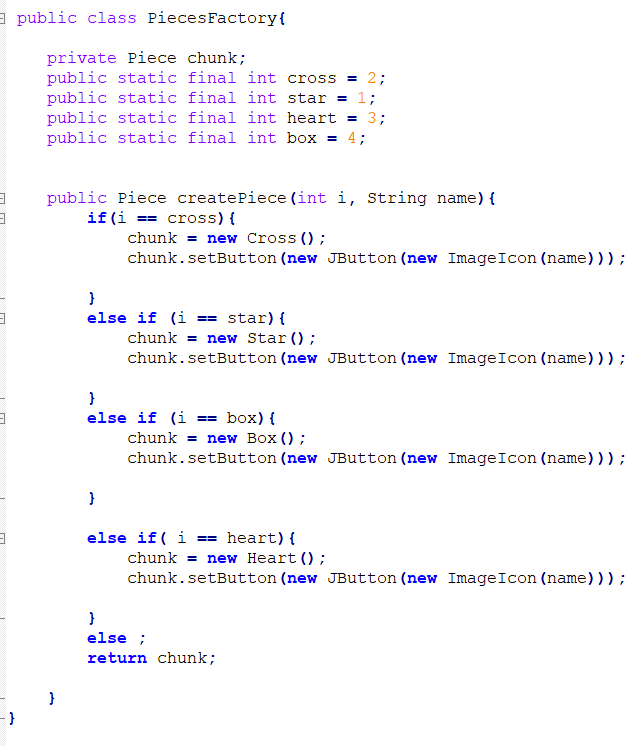


**b. Abstract Factory Desing pattern.**

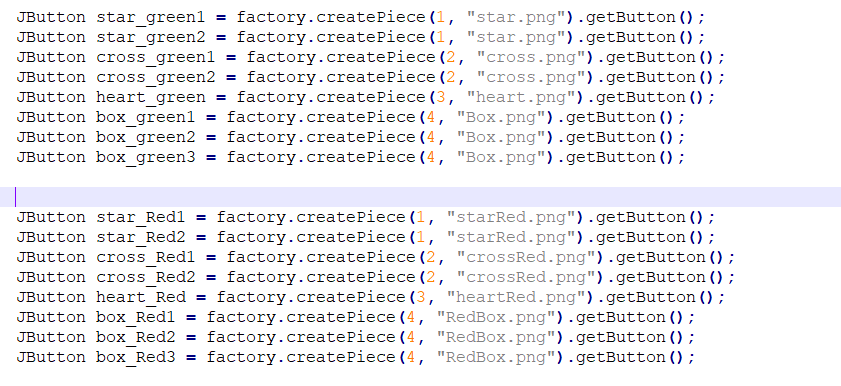
Abstract factory patterns work around a super-factory which creates other factories. This factory is also called as factory of factories. This type of design pattern comes under creational pattern as this pattern provides one of the best ways to create an object.

Abstract Factory design pattern is applied in our game on the piece objects creation in the Board class.

The implementation of PiecesFactory class which contain cratePiece method the work in object creation as follow.



The implementaino of PieceFactory method to create abjects of type piece is doen as shown in the picture below.

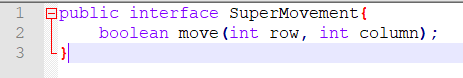


1. **Strategy design pattern**

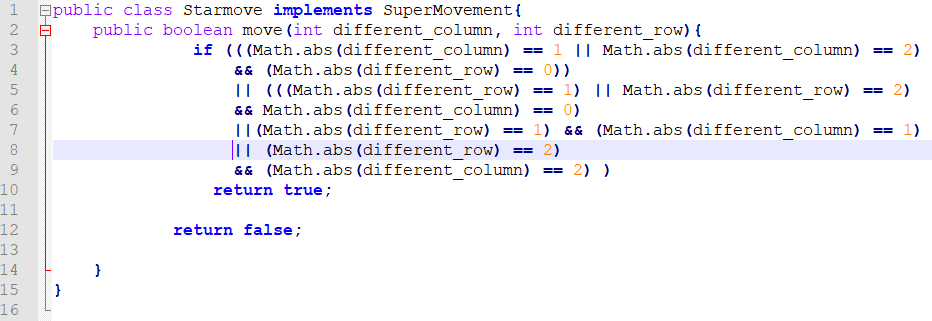
the strategy pattern is a behavioral software design pattern that enables selecting an algorithm at runtime. Instead of implementing a single algorithm directly, code receives run-time instructions as to which in a family of algorithms to use

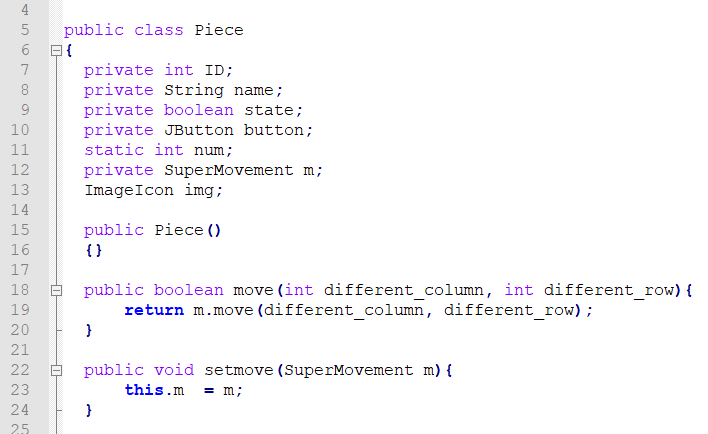
Strategy design pattern is applied in our game to encapsulate the behaviour of each pieces. The piece objects’ behaviour could be changed dynamically during runtime.

The follwing pictures illustrate the implemetation of Strategy desing pattern.



This step has been implemented for Starmove class, Heartmove class, Boxmoveup class, BoxmoveDown class , and Crossmove class.



In the code bleow the variabes and methods (private SuperMovement m), (public boolean move()), (setmove()) is to implement the Strategy dessing pattern.  


The following code illustrate how the piece can change its movement using startegy dessing pattern

