STREET SHOOTOUT

*Adeel Ahmed [321844]*

*Syed Umer Tariq [304686]*

*Sheikh Mustafa Ahmed [305737]*

*End Semester project OOP*

School of electrical engineering and computer science, SEECS

**Abstract—**

**Street Shootouts is a 2D-penalty game for appeasing the crave of football enthusiasts. The game delineates a penalty shootout in which user plays as penalty taker and has to put the ball inside the net by beating the keeper .For scoring each goal the user gets points unless the keeper makes a save which ends the player's turn which insinuates the Death Penalty scenario. This game provides you an experience of what it is like to play Death Penalties. Now you are ready to get on with the game. Just beat the keeper as many times as you can to score maximum points!!! Challenge your friends to beat your score!!!**

**Introduction—**

The game is built on the javafx class to provide the GUI for our game this provides a basic outlook of the project while the components include the four basic pillars of object oriented programming that includes inheritance, polymorphism, encapsulation, and abstraction, most abundantly used was inheritance which was applied in most of the classes that were in use in the project.

The main options for the game and how to play it is that the penalty taker via the user can decide in which direction those he or she wants to put the ball in and at that position the football moves into place the point is decided by the user but the goal keeper may move randomly that is the luck of the user weather it is blocked or not the goal keeper moves arbitrary along any of the given directions that are present for it.

The user may provide a direction but the goal keeper may randomly move in any direction if the direction of the user and the arbitrary selected direction point of the goal keeper coincide then the goal will be stopped and the game will be over and your score will be saved but at the start only the high score will appear whom so ever it may be the score are saved via file handling the player may choose whether to quit the game in between or resume it however he may like it.

The main idea of the game is to provide a thrill for those who may wish to enjoy football as the penalty shootouts of any game is thrilling and inspiring for those who have interest in the game the main theme used for the game is that of a localized penalty shootout scene in which the user may see only the keeper the goal post and net, the only thing between him will be the goal keeper trying to defend his goal post from the player.

The theme for the player is quite mesmerizing that shows a look out of the local street art and along with the penalty box being placed in between in which the user’s penalty taker and the goal keeper the idea of this game is to promote the local art and the game’s most exquisite part therefore the game is most likely to bring along both sides of the argument the game also provides a voice media for more realistic approach the user may also if he wants pause the game or even may quit if he feels so.

The game as in the description that will be presented bellow shows the working of this project and the calamities that present inside of it and the usage of the concepts that we learned over a period of time all of which are included in the project and also explained in detailed in the project report.

The project with the main purpose of defining the penalty shootout game was not built as basic but rather include some things that needed description therefore the project report will explain it put the controls of this game are rather simple

**Tool’s used—**

The primary tool used for this project is the java programming language but the IDE used is Intellij and along with that for including and supporting the goal keeper and the penalty taker and there graphics in our project we used Adobe Illustrator.

**Motivation—**

All three members of our group are football fans and have enjoyed football as a past time therefore both playing and watching football has motivated us to build this project and fulfill the need to play football and most importantly penalty shootouts. Because they are the most sophisticated part of the game therefore was our decision to make a project or rather a game that may satisfy the need for playing and enjoying this game

**Method—**

For the purpose of explaining all the concepts and programming technique used in this project we divided the project description into different classes and each individual class there will be the completed detailed analysis of what type of Object Oriented concepts are used in these classes are given step by step and the name of the classes along with the properties they hold and the concept’s used are all given bellow as the following

**Application Class:**

Application class is the base class for making the whole project as it is extended by all the main classes in the project and in this way the whole project implements the concept of abstraction.

From this class our whole project gets the base essence of one of the prominent principle of OOP which is "Abstraction". So Application class has utmost importance in this project as our project deals a lot with Javafx.

**MainScreen Class:**

MainScreen class is a public class used to create main screen for the game which is when you start the game it is the class which is first loaded. This class acts as the back bone of the project from where all things start and this class has the functionalities to deal with how the game starts that is when the game starts so the loading screen appears with the logo of the game which has been defined in this class and this class asks for user response to move the user to the "Main Menu" of the game.

This class extends the Application class which is an abstract class and defines its start method. This class incorporates the features of both javafx and java in terms of using their libraries.

MainScreen class implements basic principles of OOP such as Inheritance, Encapsulation.

Inheritance is implemented in the way that this class extends Application Class of Javafx and implements its start method.

All the data fields of MainScreen class have been made private in order to encapsulate them and prevent them from being called from outside the class.

**MainMenu Class:**

MainMenu class is a public class used in the project for creating the Main Menu of the game. This class contains all that takes for the main menu of the game. Main menu is one of the things that plays an

important role in making a game charismatic so this class has its importance in that way. Further on this classimplements thefunctionalities which a main menu for any game has that is this class shows the user different option as seeing controls of the game, game description, watching your highscore, exiting the game and most importantly to access the option of starting the game which takes the user to the actual game playing scenario.

This class extends the Application class which is an abstract class and defines its start method. This class incorporates the features of both javafx and java in terms of using their libraries.

MainMenu class also interacts with the basic principles of OOP like Inheritance and Encapsulation.

Inheritance is implemented in the way that this class extends Application Class of Javafx and implements its start method.

All the data fields of MainScreen class have been made private in order to encapsulate them and prevent them from being called from outside the class.

**MenuOptions Class:**

MenuOptions class is an abstract class that is used to create concrete classes namely Controls, Highscore and GameDescription. This class is used for the implementation of the classes of the main menu options and contains all the attributes and functionalities which are common in its children classes. So this is the base class for all the menu options classes.

This class implements the core concept of OOP in the form of abstraction.

This class extends the Application class which is an abstract class and in addition this class incorporates the features of both javafx and java in terms of using their libraries.

Inheritance is implemented in the way that this class extends Application Class of Javafx and implements its start method.

**Controls Class:**

Controls class extends the abstract class MenuOptions. This class deals with the controls option of the main menu and provides its functionalities that is when the user clicks on Controls option in the Main Menu of the game so this class comes into action and takes the user to a different page which contains the informationrequired toacquaint the user with the controls of the game like how basically the game is played.

This class extends the MenuOptions class which is an abstract class and defines its abstract method.This class incorporates the features of both javafx and java in terms of using their libraries.

Controls class implements the core concepts (principles) of OOP in the form of Inheritance and Polymorphism.

Inheritance is implemented in the way that this class extends MenuOptions class and performs overriding on its "styling" in order to incorporate Polymorphism**.**

**HighScore Class:**

HighScore class extends the abstract class MenuOptions. This class inculcates the feature of highscore in the game by keeping record of users score and determining whether it is the highest score made by the user. This class is connected closely with the streetShootouts class in terms of its relations with highscore variable of that class.

This class extends the MenuOptions class which is an abstract class and defines its abstract method.This class incorporates the features of both javafx and java in terms of using their libraries.

HighScore class implements the core concepts (principles) of OOP in the form of Inheritance and Polymorphism.

Inheritance is implemented in the way that this class extends MenuOptions class and performs overriding on its "styling" in order to incorporate Polymorphism.

**GameDescription Class:**

GameDescription class extends the abstract class MenuOptions. This class has been made to give user a little bit of know how about the game what the game actually is and what is the concept behind this game and clears all the doubts of the user about the game so the user can be very clear about the game actually is.

This class extends the MenuOptions class which is an abstract class and defines its abstract method.This class incorporates the features of both javafx and java in terms of using their libraries.

GameDescription class implements the core concepts (principles) of OOP in the form of Inheritance and Polymorphism.

Inheritance is implemented in the way that this class extends MenuOptions class and performs overriding on its "styling" in order to incorporate Polymorphism**.**

**GameEndIndication Class:**

GameEndIndicaion class is a public class which sole purpose is to deal with the particular scenario of when the game gets over that is when goalkeeper makes the save and user's turn gets over so this class has the functionalities related to previously mentionedcharacteristics and shows the user it's score and provides the user with the options of whether to play the game again or move to main menu.

This class extends the Application class which is an abstract class and gives explanation to its start method. This class incorporates the features of both javafx and java in terms of using their libraries.

GameEndIndication class implements the concepts of OOP namely Encapsulation and Inheritance.

All the data fields of MainScreen class have been made private in order to encapsulate them and prevent them from being called from outside the class.

Inheritance is implemented in the way that this class extends Application Class of Javafx**.**

**HighScoreManaging Class:**

HighScoreManaging class is a public class whose responsibility is to save the highscore of the user in the file. This class implements the File Handling in a way that the high score made by the user is saved in the file and then this class is called in different classes in order to retrieve the high score from the file.

All the classes in the project are very deeply related with each other in a way that all the classes in some way call each other so it can be a perfect example of "HAS-A" relationship which strengthens our use of classes and objects which is the main purpose of any object oriented programming like the one we are working on this project which is JAVA.

**StreetShootouts Class:**

The streetShootouts class is a public class containing all the necessary fields, data and methods to run the game mechanism.

* Fields in streetShootouts Class:

dLeftBT (Button):

Button that triggers the program resulting the shot to be directed at bottom left corner.

uLeftBT (Button):

Button that triggers the program resulting the shot to be directed at top left corner.

uRightBT (Button):

Button that triggers the program resulting the shot to be directed at top right corner.

dRightBT (Button):

Button that triggers the program resulting the shot to be directed at bottom right corner.

upBT (Button):

Button that triggers the program resulting the shot to be directed at upward direction.

downBT (Button):

Button that triggers the program resulting the shot to be directed at downward direction.

randomGK (int):

Stores the random integer generated using the getRandom method.

Option (int):

Stores an integer as an option selected by the user.

Score (int):

Stores the current score of the game scored by the user.

hScore (int):

Stores the high score of all games.

man (Group):

A node that displays all the imageviews of the player sprites.

ballGroup (Group):

A node that displays the ball on the stage.

GK (Group):

A node that displays all the imageviews of the goal keeper sprites.

groundgrp (Group):

A node that displays the ground on the stage that show the layout in which the user is shown the land that is marked by yellow lines that indicate the penalty box demarcation for the user therefore giving a viable understanding of the ground.

* Methods in streetShootouts Class:

getRandom(int min, int max) method:

This method generates a random integer between the maximum and minimum bounds. For example, getRandom(1,7); (minimum=1 and maximum=7) generates a random integer between 1 and 6 inclusively.

diasableButtons() method:

This method disables all the onscreen buttons that were assigned for ball movement direction and user choice of kicking the ball at a specified direction. This method also sets the opacity of these buttons to 0 for better user experience.

enableButtons() method:

This method reverses the effects of disableButtons() method and enables all the choice buttons with no change to button opacity.

getScore() method:

Getter for the score field

start(Stage stage) method:

This is an abstract method of the Application class and it basically starts the application by setting the scene and showing the stage.

Classes used in the start() method:

**Timeline Class:**

This class inherits from animation class, and the purpose of the given class is this class to animate our sprites.

The animation is done using KeyFrames as the Timeline processes individual KeyFrames sequentially in the order specified by the KeyFrame.time. A loop or cycle can be made using the KeyFrames for loop animations.

**TranslateTransition Class:**

This class inherits from Transition class, and the purpose of this class is to translate our animations made using Timeline class spanning over given time period.

**SequentialTransition Class:**

This class is used alongside **PauseTransition** Class to delay the start of translation to sync other transitions with each other.

**Group Class:**

A Group node contains an ObservableList of children that are rendered in order whenever this node is rendered. It is used to display nodes on the stage. Any effect(such as resizing an image) can be applied to all images in a group.

**Results—**

When the game is started it prompts its main screen showing four plausible options of the main menu the description and the main game starting point and the control game rules the game when started shows a wall in which the goal post is placed along with the art work in the back ground to the outlook of a street shootout the game then starts by giving the user who is the penalty taker he is allowed a number of options along which direction the user may want to put the ball behind the post for that the user may choose any one of the given choice therefore the user picks one via pressing the button.

After pressing the button by the user the shot is prompted to the position of the users choice after that the goal keeper who is present between the user’s penalty taker and the goal dives in any of the giving position but the dive of the user is arbitrary that means it may dive in any direction therefore if the direction of the user’s penalty taker and the goal keeper’s dive is different that the goal will count and the points will be given to the user.

If the goal keeper dives in the same direction as the user’s decided position for the shot then the goal will not stand and the user’s game will be over therefore his score will be saved in a file if the score is the maximum then all the past participants then the score will be held in the high score column.

After the goal is saved the screen prompts game over and then moves back to the main menu in which the user may play the game all over again.

This game provides a point of inference and is purely based on luck whether the goal keeper arbitrary chooses the same spot as the user the game will be over due to the goal being saved, due to which the user may if they want to may play again the game also provides a unique form of thrill you may not know until the final point whether the goal is saved or not.

The game also provides voice media that allows the user to indulge more into the game, the user may also find that realistic to the main part of the sport in the real world.

**Discussion—**

The project was for the most part build on the javafx class meaning a work of GUI that made the project more viable to the user as a user friendly interface the javafx class along with the use of Adobe illustrator were used to define the paramount and observant part of the game the user had access to the games control system and the description of the game which are both present in the main menu.

The project also includes the four basic pillars of object oriented programming that includes inheritance polymorphism abstraction and encapsulation the main usage of this was for different situation but inheritance was used abundantly in many of the differentiated classes therefore the most important part of object oriented concept used for this game.

Polymorphism was also used in some parts of the project to indicate the same task differently implemented therefore the use of it and abstraction was used mainly in the application class therefore along with the javafx class with it , the less applicable but used point of object oriented programming was encapsulation which was used.

The main reason for this program to be productive and being able to be enjoyed is that the basic screen and stage fluidity and long with the voice media present that allows the person to be more connected with

the game therefore it is more viable and provides a lot of fun for the user.

After the game is over the user is prompted that on the screen the therefore returns it on the main menu therefore the user may play the game again if the score of any user is greater than of all in the past then the user may see the score prompted as the high score, until another player may beat that score the pervious score remains the highest.

The idea of high score also creates a type of completion as the user’s may compete with one another turn by turn due to being a single player game and compete with them therefore the game will be of frequent use and allow the person to be more indulge with it the user’s may also find the play to be better when he sees the iconic view at the back.

At the back of the goal post it shows the abstract and artistic view build on the basis of adobe illustrator therefore it is more practical as it is a street shoot out rather than being formal as a real game it has the looks that indicate the more ways in which football penalty shoot outs take place in the streets that also says about the logical connectivity of the title of this game and also the graphics that mirror the image of what we desire form the game and plan to show to the user.

**Conclusion—**

This java based program which is built to show a penalty shootout game as our end semester project is based mostly on the basis of graphics that are built on the javafx class that allows the user to prompt a screen and its attributes therefore allowing us to display the ground that is mostly green other than that having a yellow lines as demarcation which shows the penalty box and its parameters.

The goal post is directly in the wall of the art and graphics that shows points of buttons of the selection for the user therefore they are shown there. The user commands the penalty taker and the goal keeper moves arbitrary therefore the player may choose any point in which they would want to go but the goal keeper will dive in any direction every time but the user may only loose if the position of the goal keeper dives in the same direction as the football the goal may be saved and the game will be over.

The football has its own attributes and its functions are the same in every task only the point at which to move when decided by the user may change otherwise the ball will move in the direction as given by the user the penalty taker will move in the same way but the ball will move in the direction selected by the user and follow the designated path that is attributed to the different given directions.

**UML DIAGRAM:**

****