Group Member:

Zhengsheng Li(216193443) Zihao Wu(214808455) and Siqin Zhang(214838023)

Course: EECS3311

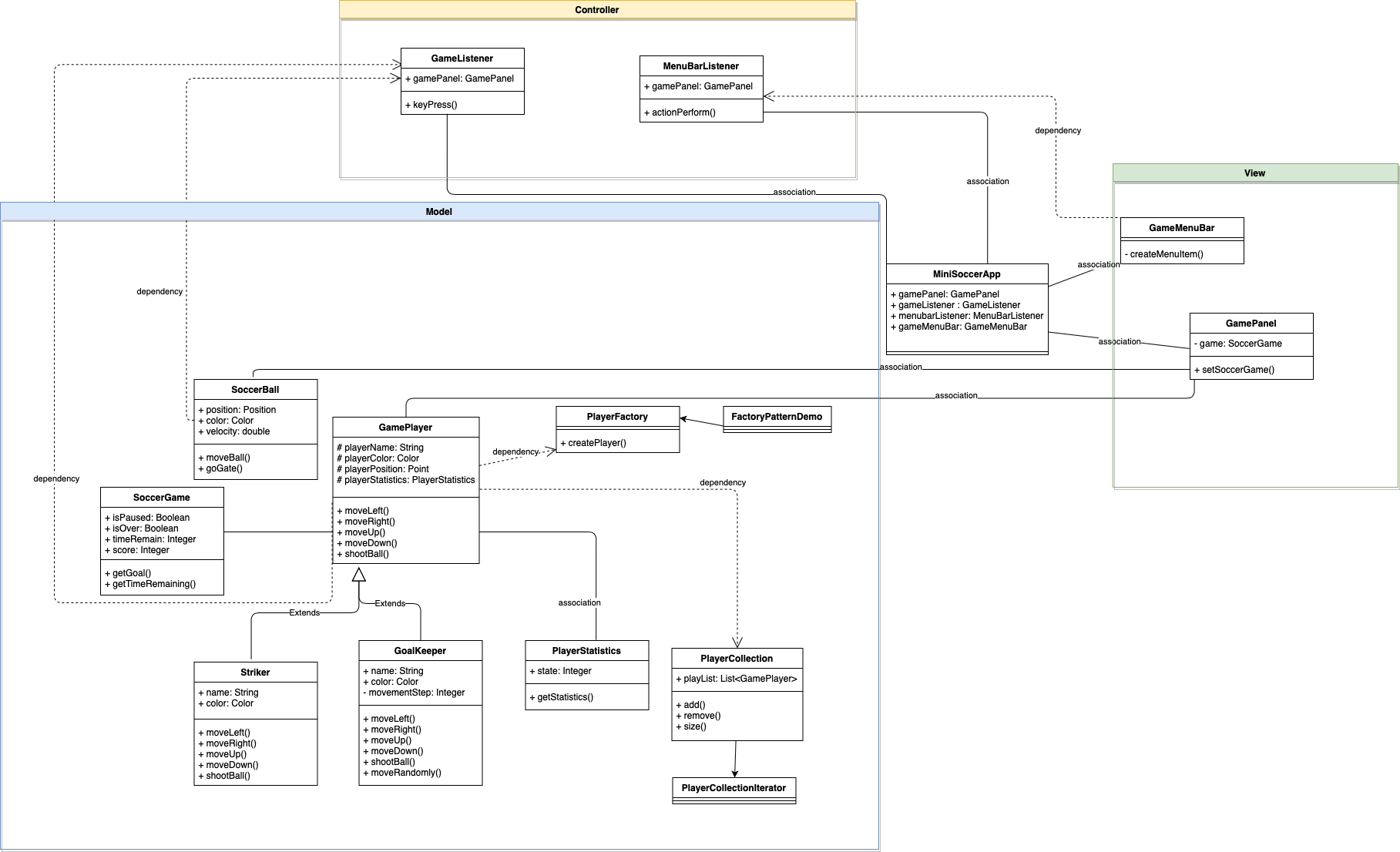
Section: B

Software Project: Display Shapes

TA’s name: Naeiji Alireza

Part I:

1. The project is about creating a game called mini-soccer which aims to make some scores within 60 seconds.
2. The issue met is the association between player and ball. The player has to dribble the ball wherever he moves until he shots.
3. The project will use the knowledge of OOD, which includes states, behaviors and interactions. When we run the program, it is in a Pause state initially. In this case, neither the game player nor the soccer ball can move. When the time becomes 0, it is in a state of Game Over, and in this situation, the pause and resume buttons would never use unless a new game is started. During the game, the goalkeeper will move randomly around the gate, which shows the behavior of the goalkeeper, and when the player shots the ball outside the gate or in front of the goalkeeper, the goalkeeper will catch the ball and kick back to the side of the sticker, which shows the interaction between Player and Goalkeeper, Player and Ball, as well as Goalkeeper and Ball.
4. We will first analysis the project, then we know how to create an UML diagram by showing the relationships, and then implement the project based the diagram.

Part II:

a.

It is found that the diagram can be split into three parts: Controller, View and Model. Model responds the request from the Controller, and the View displays the data in the Model. The Controller is responsible for handling the interaction between View and Model. In this diagram, GamMenuBar and GamePanel are defined as View, whereas GameListener and MenuBarListener are defined as Controller. The Model includes GamePlayer, Goalkeeper, Striker, and other relevant classes that function on GamePlayer.

b.

The patterns used in the diagram are the Factory pattern and the Iterator pattern.

The Factory pattern is to create objects that is referred to using a common interface. In this case, game players would be the interface, and the striker and goalkeeper would be objects. By the FactoryPatternDemo, the PlayerFactory would create Striker and GoalKeeper, which are referred to GamePlayer.

The Iterator pattern is used to iterate over a method. From the diagram, it is showed that PlayerCollection will be iterated by PlayerCollectionIterator.

c.

The OO design principle applied in the project is inheritance and encapsulation, For the inheritance, child class will inherit parent class’ method. From the diagram shown above, it is obvious that both Striker and GoalKeeper are inherited from GamePlayer. While they can do same as the GamePlayer, they have their own behaviors. For instance, the GoalPlayer has the behavior of moving randomly and catch the ball, and Striker would have the ability of dribble the ball. For the encapsulation, it is the interaction between objects in different classes, like the interactions between Player and Goalkeeper, Player and SoccerBall, Goalkeeper and SoccerBall, etc.

Part III

GamePlayer: any player of the soccer game (e.g., striker, goal keeper)

Goalkeeper: it comprises the goal keeper’s attributes and operations

Striker: it represents the striker’s attributes and operations

PlayerStatistics: it indicates the statistics of a player (i.e. caught balls, or scored goals) PlayerFactory: factory that creates players

PlayerCollection: an iterable Collection of players that needs to have methods to initialize a collection of players, to add a player and to remove a player from the collection

Implementing and compiling: Firstly, I have create all missing classes and functions to remove the warnings, followed by implementing all blank functions. However, I did not write the PlayerCollectionIterator class because I am not sure what PlayerCollectionIterator for. Actually, I have use the interface java.util.Iterable and override the functions that was needed.

The tool I used on this project is Eclipse IDE 2021-09 finally, but I have used the one on VirtualBox from the department, and JDK is JDK 16 version.

PartIV:

In conclusion, it would be a successful project although it is a kind of easy project. Firstly, the creating player part, players classes went well once it implemented but I tried lots of time for the playerStatistics class to fix the goals and caught number of balls because it was always show me an address of the object but not the number of it. The basic parts is easier to implement which include gameplayer class, factory class and playsercollection class. The one that need to be careful is use on Interger as an object instead use int directly and displaying correctly.

The advantages of completing the lab in group is that I could easier to know about how the project work and getting ideas of building project structure because we could get the good ideas from others and give up wrong thinking we did before. The disadvantages would be we are in different time zone, it is hard for us to book a time for meeting.

We are 3 member group who are ZhengshengLi(216193443), Siqin Zhang(214838023) and Zihao Wu(214808455). Firstly, we discuss the whole project (Zihao Wu gave the idea about how to design and report partIV, Siqin Zhang wrote the code and report partIII, Zhengsheng Li drew the diagram and report partI&II)and give each other some advices about the mini game. After writing, we talked about our mind and debug to imporve the project.