Environment Class

In this class there is only one method which makes the game function. This method initially creates three balls of all levels and stores them in an arraylist called “firstBalls”. Also, it creates the canvas before the while loop. Right before this always true while loop the start time of the game is identified. Within the while loop first there is an if condition that determines if there is still time and the Boolean variable that decides whether the game is on or not is true, then all animation methods from other classes are called. In a for-each loop that iterates through “firstBalls” ball-arrow and ball-player interactions are checked. If the arrow hit a ball, it is erased from the copy arraylist and two new balls are added to this arraylist according to the level of the hit ball. If a ball touched the player, the Boolean is assigned to false. Right after this for loop the reference of “firstBalls” is changed to the copy arraylist’s. The case when all balls are hit before running out of time is considered by checking whether the “firstBalls” are empty. Lastly the current time is updated. According to the difference between current time and start time, the variables are updated that are dependent on how much time has passed. Up to here all the processes have been in an if condition. After that there is an else-if condition which is true when one of the game ending scenarios comes true. Under this condition the canvas is rearranged. Lastly according to which option the user chose, the system exists, or all the variables are reassigned to their initial values.

Bar Class

In Bar Class there is only one method which displays the time bar. The variables such as the x coordinate and the width of the bar are called from the Environment class because that is where their values are updated according to the amount of time that has passed.

Ball Class

In Ball Class there is a constructor which holds the values of a ball such as its initial x and y coordinates, its level, and its speed on x axis. There are two simple methods to calculate the maximum height that a ball can reach and its radius according to its level. Lastly there is the main animation method “motion()” which contains several if checks. The first checkpoint determines whether the ball is on the right or left edge of the canvas. If that is the case, the direction of the speed on x axis is changed. The next checkpoint works under the circumstance when the ball touched the ground. If so its speed on y axis is assigned to its maximum with the physical formula (2gh)^(1/2). Lastly the speed on y axis and the coordinates are updated and the ball is printed on the canvas.

Player Class

The Player Class contains one method with two if conditions. The first condition works under the circumstance where the left key is pressed, and the player has not reached the left edge of the canvas yet. In the second condition the same logic is applied for the movement to right. According to the correctness of these conditions the coordinates of the player are updated, and it is drawn on the canvas.

Arrow Class

The first method of this class determines the activeness of the. The second one is for animation. It works when the arrow is still active and it has not reached the upper edge. The activeness of the arrow could be manipulated from Environment by making its coordinates invalid so that the arrow is inactivated. It is used for the cases when the arrow should be erased because it hit a ball.

[1]

[1] https://drive.google.com/file/d/18gEQeR3MfvdbKW2UuqzcnvEYyeZ2LqKw/view?usp=sharing