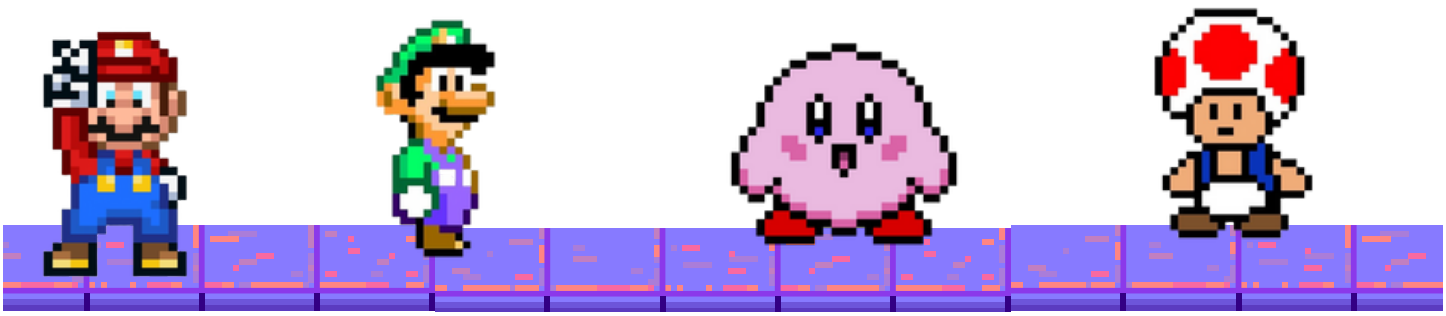
CREATORS:	NAME: JOUD ALSAYID NAME: AISHAH ALGHARIB



IN THIS GAME, A FIXED NUMBER OF OBJECTS WILL FALL TO THE BOARD FROM TOP AND MOVE DOWN AND LEAVE THE BOARD. EACH OBJECT WILL HAVE A VALUE. EACH OBJECT WILL BE FASTER THAN THE PREVIOUS OBJECT. CLICKING AN OBJECT REMOVES IT FROM THE BOARD AND ADDS ITS VALUE TO THE SCORE. IT KEEPS TRACK OF THE TOP 5 SCORES. AS A PLAYER, CLICK AS MANY OBJECTS AS YOU CAN. IT WILL DISPLAY THE SCORE WHILE PLAYING THE GAME. WHEN THE GAME IS OVER IT WILL DISPLAY THE TOP SCORES. THERE IS ONE CONDITION TO STOP THE ANIMATION IF ALL THE OBJECTS ARE PRESSED OR REACH THE BOTTOM AND THE TOTAL NUMBER OF OBJECTS IS 30.

PROJECT DETAILS

THE CLASS GAME

WORK DISTRIBUTION: AISHAH&JOOD

THIS CLASS CONTAINS MULTIPLE METHODS AND VARIABLES TO CONTROL THE SPEED CLICK GAME AND DESIGN IT. THE METHODS ARE START(), TOP5(), MAIN(), AND STOP(). IT ALSO CONTAINS ONE CLASS CALLED HANDLER WHICH IMPLEMENTS THE EVENTHANDLER INTERFACE AND OVERRIDES THE HANDLE() METHOD.

START METHOD

WORK DISTRIBUTION: AISHAH&JOOD

IN THE START METHOD, WE STYLED AND ORGANIZED THE PANES AND THE NODES. THE PROGRAM WILL CONTINUE TO PLAY UNTIL THE 30 OBJECTS GET CLICKED OR FALL. AFTER WE SET THE BACKGROUND, WE CREATED AN IMAGE FOR YOU TO START THE PROGRAM WHEN THE USER CLICKS THE START OPERATOR AN OBJECT WILL BE ADDED AND THE ANIMATION WILL PLAY. ALSO WE CREATED OBJECTS BY USING IMAGE VIEWS AND CONTROLLED THEM BY USING SET ON MOUSE CLICKED, THE OBJECTS WILL FALL AT DIFFERENT TIME. IF THE USER CLICKS AN OBJECT IT WILL DISAPPEAR AND ITS VALUE IS GOING TO BE ADDED THEN A NEW OBJECT WILL BE DISPLAYED INSTEAD WITH AN INCREASED SPEED. AND WE CREATED AN IMAGE VIEW FOR THE RESTART OPERATOR WHEN THE USER CLICKS RESTART HE WILL RESET THE SCORE AND THE NUMBER OF OBJECTS AND THE ANIMATION WILL BE PLAYED AGAIN BY

ADDING RANDOM OBJECTS AGAIN. ALSO, WE CREATED AN IMAGE VIEW FOR THE RESET OPERATOR, WHEN THE USER CLICKS RESET, THE TOP FIVE SCORES WILL BE CLEARED. LASTLY, WE CREATED A SCENE AND ADD TO IT THE BORDER PANE AND THEN ADD THE SCENE TO THE STAGE AND SET ITS PROPERTIES.

```
//overriding the start method Which normally places UI controls in a scene and displays the scene in a stage
@Override
public void start(Stage stage) {
    // Styling the border pane and vBox and the labels
    borderPane.setPadding(new Insets(5,5,5,5));
    vBox.setSpacing(3.5);
    currentScoreLabel.setFont(font);
    remainingObj.setFont(font);
    // An Animation with indefinite duration (a cycleCount of INDEFINITE) runs repeatedly until the stop() method is explicitly called
    animation.setCycleCount(Timeline.INDEFINITE);

    /* using a for loop to fill the image view array with images,
     set their properties, and set a different value for each imageview */
    for (int i = 0; i < 4; i++){
        imageView0bjs[i] = new ImageView();
        imageView0bjs[i].setFitHeight(120);
        imageView0bjs[i].setFitWidth(77);
        imageView0bjs[i].setImage(objectArray[i]);
        imageView0bjs[i].setX((int) (Math.random( ) * 650 ));
        imageView0bjs[i].setUserData(i+1);
    }
    // To make the objects fall at different times we set the y position
    imageView0bjs[3].setY(-50);
    imageView0bjs[2].setY(-180);
    imageView0bjs[1].setY(-350);
    imageView0bjs[0].setY(-500);

    // Creating an image view for the background and setting its properties, then add it to the pane
    ImageView imageViewBackground = new ImageView(new Image("backgroundImage.png"));
    imageViewBackground.setFitWidth(700);
    imageViewBackground.setFitHeight(550);
    borderPane.getChildren().add(imageViewBackground);

    // Creating an image view for the start operator and setting its properties and adding it to the pane
    ImageView imageViewStart= new ImageView("startImage.png");
    imageViewStart.setFitWidth(200);
    imageViewStart.setFitHeight(90);
    borderPane.setCenter(imageViewStart);
    // Controlling the start operator
```

```
    imageViewStart.setOnMouseClicked(e ->{
        // adding a imageView array and the labels
        borderPane.getChildren().addAll(imageView0bjs);
        borderPane.setLeft(currentScoreLabel);
        borderPane.setRight(remainingObj);
        // remove the start operator then starting the animation
        borderPane.getChildren().remove(imageViewStart);
        animation.play();
    });
```

till here work distribution by Joud.

```
    for (int i=0; i<4; i++) {
        final int index = i;
        // Controlling the clicked object
        imageView0bjs[i].setOnMouseClicked(e -> {
            // To remove the object when its clicked and add a new one
            borderPane.getChildren().remove(imageView0bjs[index]);
            // setting the properties for the new object that's going to be added with a random position
            imageView0bjs[index].setY(-50);
            imageView0bjs[index].setX((int) (Math.random( ) * 650 ));
            // setting the new object then adding it
            imageView0bjs[index].setImage(objectArray[index]);
            borderPane.getChildren().add(imageView0bjs[index]);
            // To increase the speed of each new added object
            animation.setRate(animation.getRate() + 0.9);
            // To count the number of objects that got clicked
            numOfObj++;
            // Each object has a different value that will be added to the current score when clicked
            currentScoreNum += (int) imageView0bjs[index].getUserData();
            // Updating the value of currentScoreNum in currentScoreLabel and remaining objects
            currentScoreLabel.setText("Score: " + currentScoreNum);
            remainingObj.setText("Remaining objects: " + (30 - numOfObj));
        });
    }

    // setting the properties for the restart operator
    imageViewRestart.setFitWidth(200);
    imageViewRestart.setFitHeight(70);
    // Controlling the restart operator
    imageViewRestart.setOnMouseClicked(e -> {
        // setting the properties for the objects to restart
        for (int i=0; i<4; i++) {
            imageView0bjs[i].setX((int) (Math.random( ) * 650 ));
            imageView0bjs[i].setImage(objectArray[i]);
        }
        imageView0bjs[3].setY(-50);
```

```
        imageView0bjs[2].setY(-200);
        imageView0bjs[1].setY(-350);
        imageView0bjs[0].setY(-500);
        // restarting the speed
        animation.setRate(1);
        // reset the currentScoreNum and numOfObj
        currentScoreNum = 0;
        numOfObj = 0;
        // Updating the current score number in the current score label and remainingObj label
        currentScoreLabel.setText("Score: " + currentScoreNum);
        remainingObj.setText("Remaining objects: " + (30 - numOfObj));
        // adding the objects then playing the animation
        borderPane.getChildren().addAll(imageView0bjs);
        animation.play();
    });

        // setting the properties for the top5 reset operator
        imageViewReset.setFitWidth(100);
        imageViewReset.setFitHeight(35);
        // Controlling the reset operator
        imageViewReset.setOnMouseClicked(e -> {
        File file = new File("scores.txt");
        try {
            FileWriter writer = new FileWriter(file,false);
            //clear the file to have new top5
            writer.write("");
            writer.close();
        } catch (IOException ex) {
            System.out.println(ex.getMessage());
        }
});

    // Creating a scene and setting its properties
    Scene scene = new Scene(borderPane,700, 550);
    // set the scene and styling the stage
    stage.setScene(scene);
    stage.setResizable(false);
    stage.setTitle("Speed Click Game");
    stage.show();

till here work distribution by Aishah.
}
```

Class Handelar

WORK DISTRIBUTION: AISHAH&JOOD

THE CLASS HANDLER MAKE THE OBJECTS ABLE TO FALL. WHEN THE OBJECTS REACH THE BOTTOM, IT WILL BE REMOVED AND A NEW OBJECT WILL BE ADDED. THE CONDITION TO STOP THE GAME IS WHEN ALL THE 30 OBJECT PRESSED OR REACHED THE BOTTOM.

```
// Defining a class handler to handle the animation that implements the EventHandler
interface
class Handler implements EventHandler<ActionEvent> {
  @Override
  public void handle(ActionEvent event) {
    // when the animation is played the top score and restart operator are removed
    vBox.getChildren().clear();
    for (int i=0; i<4; i++) {
      // To make the object able to fall down by updating its Y position
      imageViewObjs[i].setY(imageViewObjs[i].getY() + 1);
      // if an object reach the bottom it will be removed and a new object will be added
      if (imageViewObjs[i].getY() - 15 > borderPane.getHeight()) {
        numOfObj++;
        // the remainingObj label will be updated
        remainingObj.setText("Remaining objects: " + (30 - numOfObj));
        borderPane.getChildren().remove(imageViewObjs[i]);
```

till here work distribution by Joud.

```
        // setting the properties for the new object that's going to be added with a random x
position
        imageViewObjs[i].setY(-50);
        imageViewObjs[i].setX((int) (Math.random( ) * 650 ));
        // setting a random object then adding it
        imageViewObjs[i].setImage(objectArray[(int) (Math.random() * 4)]);
        // Speed increases throughout the game.
        animation.setRate(animation.getRate() + 0.7);
        borderPane.getChildren().add(imageViewObjs[i]);
      }
    }
    /* There is one condition to stop the animation:
      if all the objects are pressed or reach the bottom and the total number of objects is 30
    */
    if(numOfObj > 29) {
      stop();
    }
```

till here work distribution by Aishah.

```
  }
}
```

Top5 method

WORK DISTRIBUTION: JOOD

IN THIS METHOD, IT WILL KEEP TRACK OF THE TOP 5 SCORES BY WRITING ALL THE SCORES IN A FILE. AFTER THAT ALL THE SCORES IN THE FILE WILL BE READ AND ADDED TO THE ARRAY LIST (SCORES). THEN THE ARRAY LIST WILL BE SORTED WITH THE GREATEST VALUE. BY USING A LOOP, IT WILL DISPLAY ONLY THE FIRST 5 SCORES WHICH ARE THE TOP 5.

```
// Defining the top five method to add the top five scorers in the vBox
public void top5() {
   // Creating the top five label and setting its properties and then adding it
   Label top5Label= new Label("Top 5 Scores: ");
   top5Label.setFont(font) ;
   vBox.getChildren().add(top5Label);
   FileWriter fileWriter;
   // To store the scores in a file
   try(Scanner read = new Scanner(new File("scores.txt"))) {
      fileWriter = new FileWriter("scores.txt", true);
      // we add the newest value of currentScoreNum
      fileWriter.write(currentScoreNum + "\n");
      fileWriter.close();

// Clearing the scores ArrayList so that no duplicates(weren't played by the user) are added to the list
      scores.clear();
      while(read.hasNext()) {
         // To add all previous scores
         scores.add(read.nextInt());
      }
   }
   catch (IOException ex) {
      System.out.println(ex.getMessage());
   }
   // sort the scores to start with the greatest value in aim to add the top5 scores only
   scores.sort(Collections.reverseOrder());
   // creating a loop to add the top5 scores (unless the scores are not yet 5, it will add only the greatest values available in scores)
   for (int i = 0; i < 5 && i < scores.size() ; i++) {
      Label top5ScoreLabel = new Label("" + scores.get(i)) ;
      top5ScoreLabel.setFont(font) ;
      vBox.getChildren().add(top5ScoreLabel);
```

```
    }
}
```

Stop method

WORK DISTRIBUTION: AISHAH

THE MAIN FUNCTION OF THIS METHOD IS TO DISPLAY THE TOP FIVE SCORES AND THE RESTART OPERATOR AND STOP THE ANIMATION.

```
// Defining the stop method to show the top five scores and the restart operator and stop the animation
public void stop() {
    // to remove the object from the pane then stop
    borderPane.getChildren().removeAll(imageViewObjs);
    animation.stop();
    // calling the top5 method
    top5();
    // adding the restart operator
    vBox.getChildren().add(imageViewRestart);
    // setting the vbox properties then adding it to the border pane
    vBox.setAlignment(Pos.CENTER);
    borderPane.setCenter(vBox);
}
```

Main method

WORK DISTRIBUTION: JOUD

THE MAIN METHOD IS USED TO LAUNCH THE PROGRAM

```
// The main method to launch the program
public static void main(String[] args) {
    launch(args);
}
```

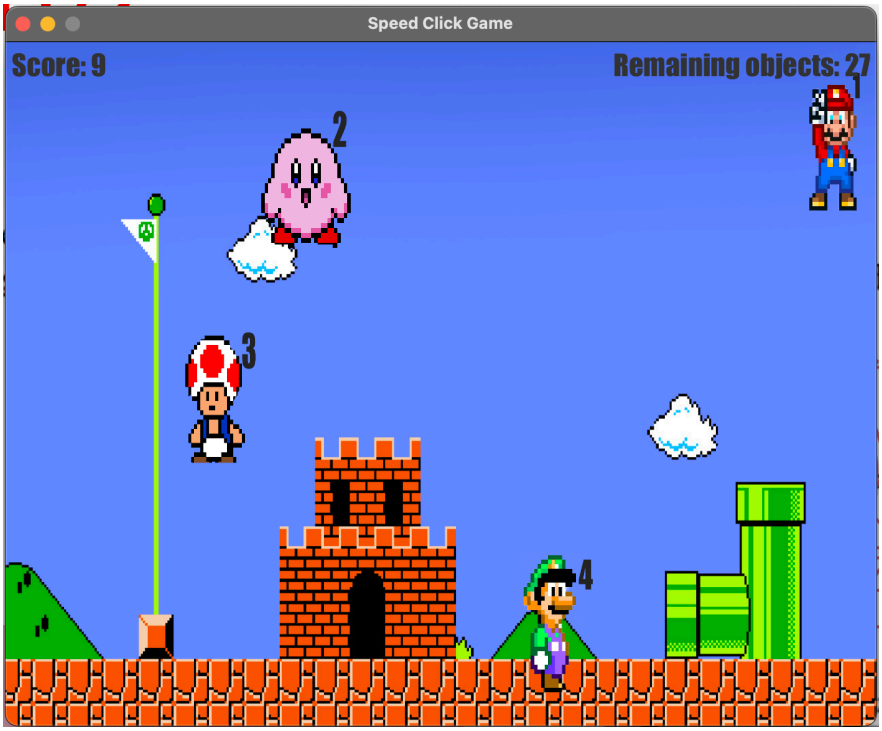
- ***Game starts when the user clicks starts***

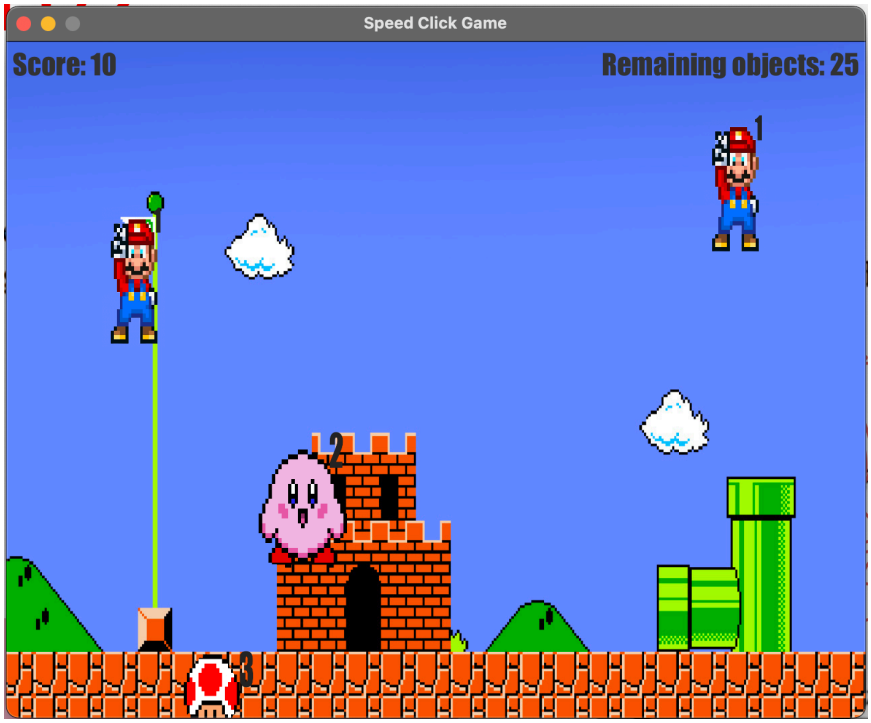
- ***The user should click on the object to increment his score***


Speed Click Game
Score: 0
Remaining objects: 30
3
4

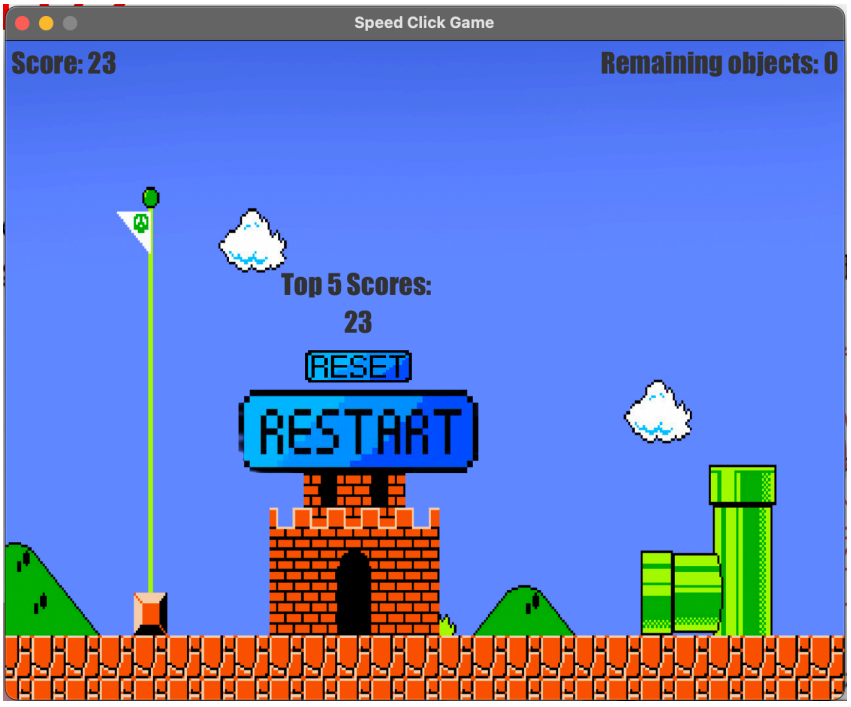

Speed Click Game
Remaining objects: 29
4

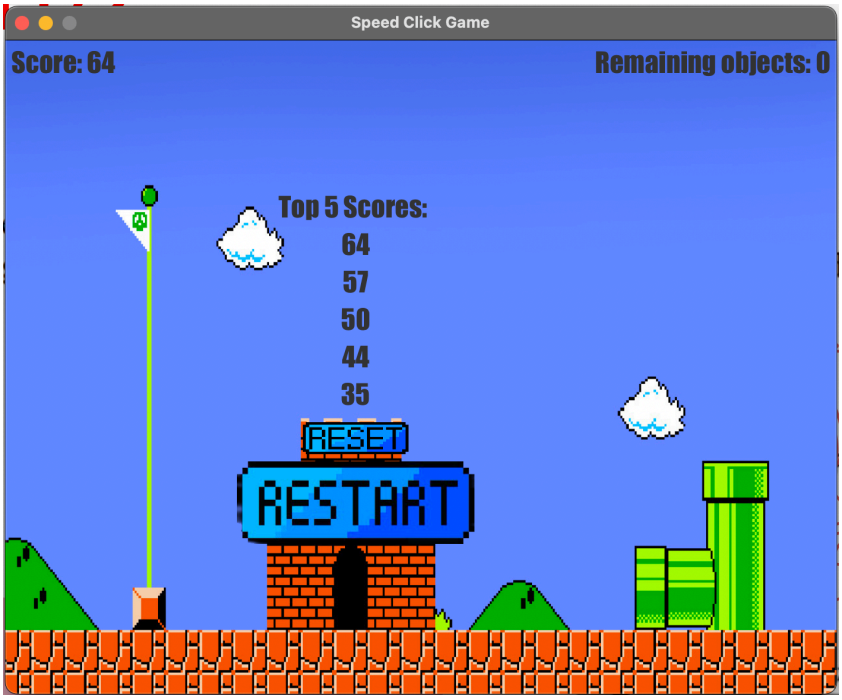

Speed Click Game
Score: 7
Remaining objects: 28
2
3
4


Speed Click Game
Score: 9
Remaining objects: 27
1
2
3
4



- ***The top 5 scores will be displayed at the end of each round***





- ***when the user clicks on reset, the top 5 scores will be cleared and only the score of the new round is displayed***