AP – PROJECT

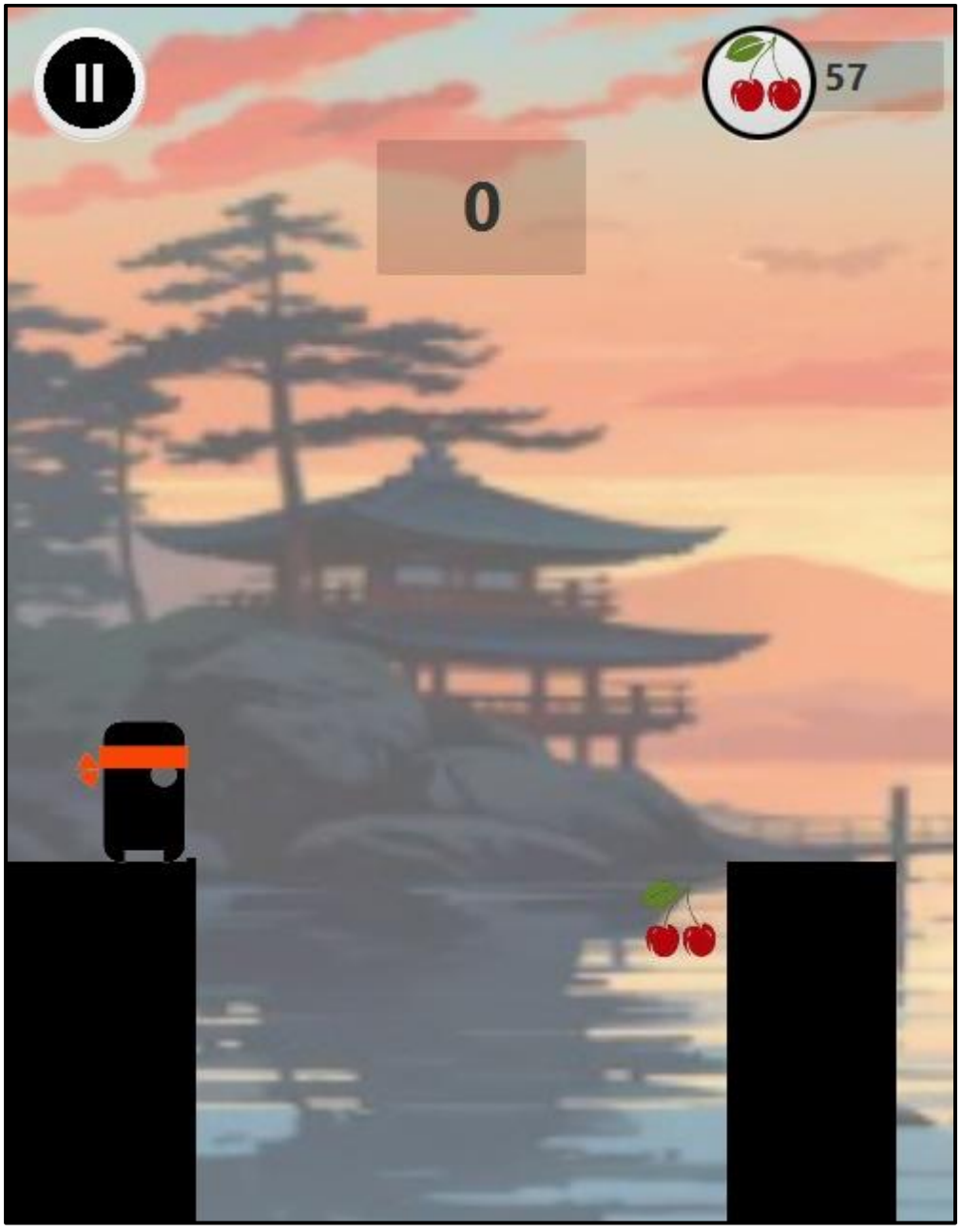
Stick Hero Game

Group Members:

1. Aditya Gupta (2022031)
2. Debjit Banerji (2022146)

Implementation Details:

1) For the stick-hero character, we have used the same image as in the real game. The character will initially have a stick of length 0 when it reaches any platform. The user can then make the stick long by holding down the left click of the mouse. After the user releases the mouse, the stick will fall down (rotate by 90 degrees) and will join the current and the next pillars (if the length of the stick is appropriate).



2) The game will keep generating multiple pillars of different width, with different distances between each pillar, all of which will keep getting stored in an Arraylist of Rectangles.

3) When the user does not elongate the stick enough or elongates it more than required, the stick-hero character will fall down into the abyss, producing a dying sound, followed by the "Game Over" screen.
   The "Game Over" screen contains the information about the Highest Score of the user, the current score that the user had before dying, and it gives the user three options:
   i) Return to Home Screen:
      This option allows the user to return to the home page of the game.

   ii) Revive:
      This option allows the user to revive his fallen stick-hero to continue the game from where the character died the previous time with the same score.
      When this option is clicked, the user will be shown a dialog box confirming the revive option which will cost the user 10 cherries. If the user will have lesser than 10 cherries, then the revive option won't be available to the user.

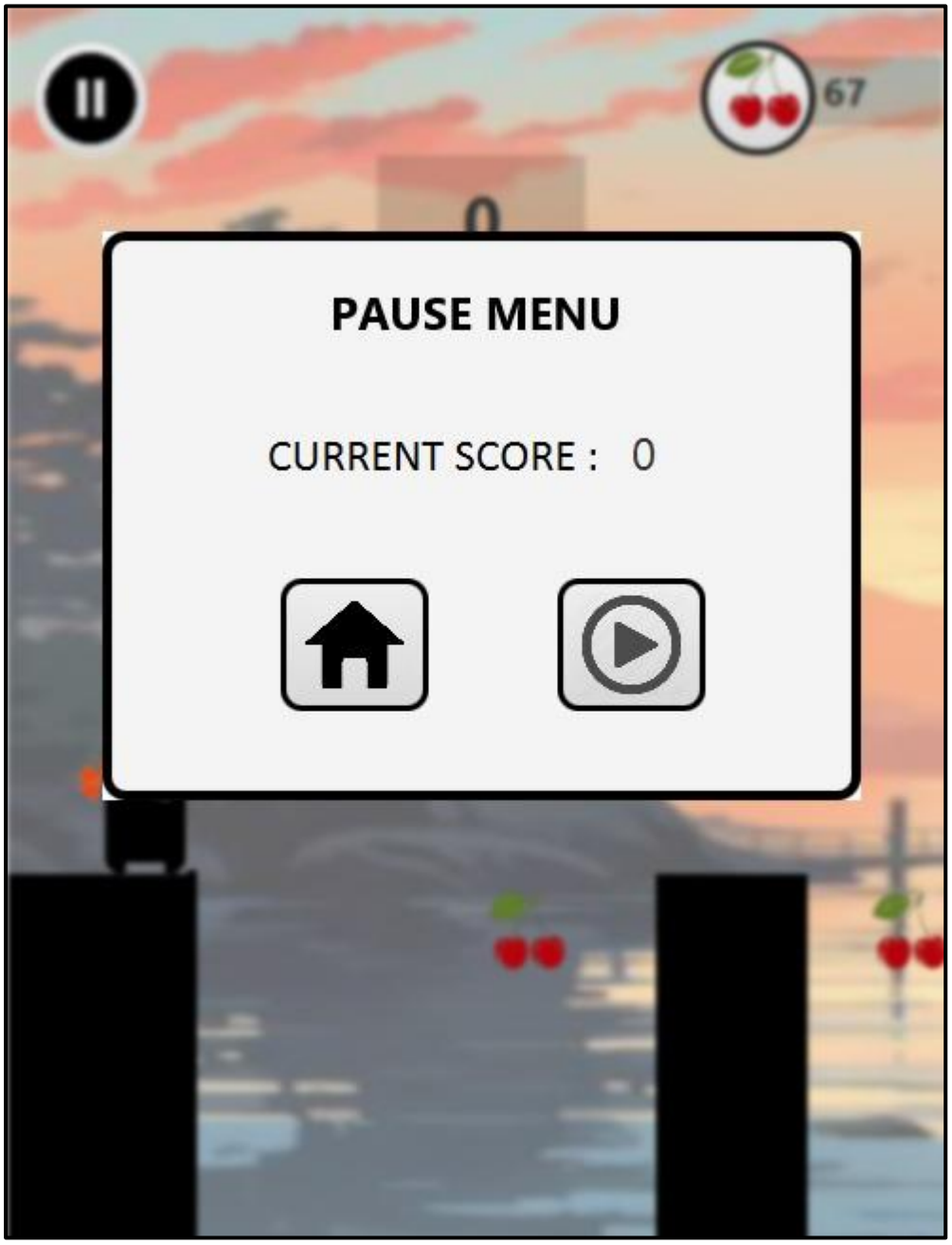
   iii) Restart:
      This option allows the user to restart the game from the beginning, with the initial score as 0.

4) When the user does not elongate the stick sufficiently or elongates the stick more that required, the stick hero will travel the distance of the stick and then fall into the abyss with a dying sound, which will result in the “Game Over” screen popping up described above.

5) The game will randomly produce cherries between any two towers, at any random position between the two towers. The user can flip his stick-hero character to collect the cherries by pressing the “S” key. The user can strive to collect more cherries, to use for revives and keep increasing his/her high-score.

6) The game will have a scoring system which will award 1 point for every pillar the stick-hero character reaches. When the user sets a new high score, after the stick-hero character dies, the game will show that a new high score has been reached and will congratulate the user using a celebratory sound effect. The user can try to set a new high score by trying to reach as many towers as possible.

7) The game will save the current score of the player as soon as he hits the pause button ((located at the top left corner of the screen). The game will also save the highest score and the total cherries of the user when the user pauses the game or the stick-hero character dies by falling into the abyss. If the user pauses the game, and decides to return to the home screen, the total cherries and his latest score will be saved. In addition to this, if the user had beat his high score and set a new record, that will also be saved.



8) The game has a background music for the Home Screen, and the Running Game screens. It also has sound effects for when the stick-hero reaches a platform, dies by falling into the abyss, collects a cherry or when the user sets a new high score. The game also has interactive menus, good graphics and an aesthetic background image for the running game screen.

Notes:

- We have implemented 2 design patterns which are:
    - Decorator design pattern for storing and reading the values of the user's "latest_score", "highest_score" and "total_cherry_count" to and from the file "PlayerState.txt"

```java
public static ArrayList<Integer> getPlayerState() throws ClassNotFoundException , FileNotFoundException {
    ArrayList<Integer> value_string = new ArrayList<>();
    Scanner in = null;

    try {
        in = new Scanner(new BufferedReader(new FileReader( fileName: "AP Project/src/main/java/project/PlayerState.txt")));
        while (in.hasNext()){
            int c1 = Integer.parseInt(in.next());
            value_string.add(c1);
        }
    }
    finally {
        if (in != null){
            in.close();
        }
    }
    return value_string;
}
```

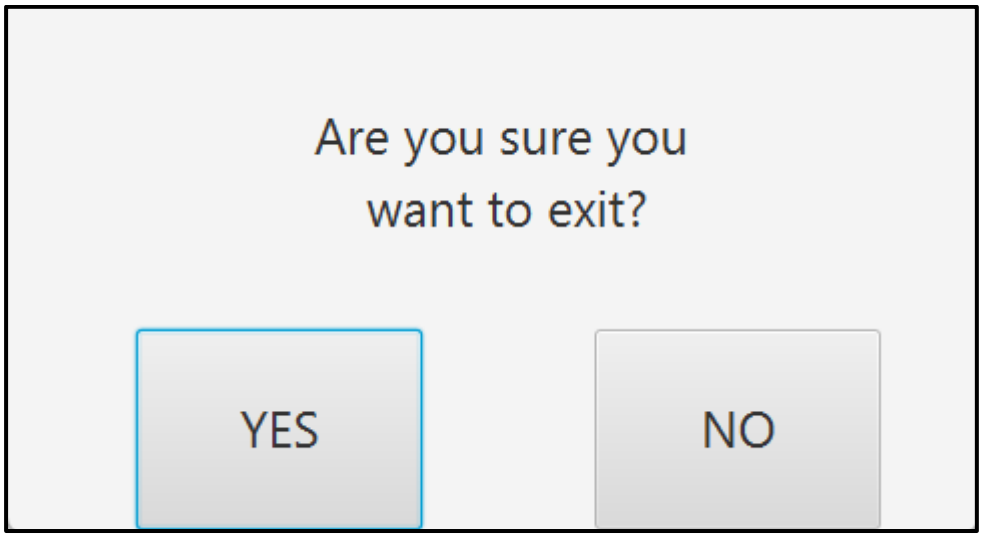
- Iterator design pattern for iterating over all the cherries stored in the “cherry_array” while the game is running.

```java
Iterator iter = cherry_array.iterator();
int i = 0;
while (iter.hasNext()){
    Cherry obj = (Cherry) iter.next();
    if (obj.getX() < Platforms.get(current_Platform+1).getX() && Platforms.get(current_Platform).getX() + Platforms.get(current_Platform).getWidth(
        cherry_counter =  i;
        break;
    }
    i++ ;
}
```

- We have implemented threading by creating a Runnable type object for the shifting the stick-hero character and the pillars of the game when the stick-hero character moves forward. We use this runnable type object to perform this function by creating and running it as a separate thread.
- The game needs to be played by clicking the green coloured “Play” button in IntelliJ Idea in the “StartApplication.java” file.
- We have used FILE I/O for input and output into the “PlayerState.txt” using Serializable Interface.
- We have also used many OOPs concepts in our game implementation such as “RodInterface” and “HomeInterface” to ensure that the respective classes implement the basic functionalities required for the object and the screen respectively.
- We have also implemented inheritance by extending the “HomePageController” in other classes like “EndingScreenController” and “PauseMenuController”.
- We have also used the concept of Composition by creating an object of type “Cherry” inside the “Player” class.

Home Screen:

- When the user clicks on “Exit” on the Home Screen, the game will open a dialog box asking the user to confirm that he/she wants to exit the game.



References:

- Fortnite Song (https://downloads.khinsider.com/game-soundtracks/album/fortnite-battle-royale-soundtrack)
- Background Image, Stick-hero images of Home Screen and Running Screen from Google
- Sound Effects from the internet

GitHub Repository Link:
https://github.com/Debjit-Banerji/AP_Project