

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

RELAZIONE DI JUMPING BALL

Si comincia con il riportare la descrizione del gioco che abbiamo realizzato:

Titolo Progetto:	Jumping Ball	
Classe:	4B ITT a.s. 2023/2024	
Nome del gruppo:	Galaxy Java Development Team	
Componenti del gruppo:	<u>Giuseppe Carlino</u>	
	Elia Grandi	
	Sousane Souhaib	
Specifiche tecniche:	Linguaggio (con versione):	Java (JDK 21)
	IDE (con versione):	Apache NetBeans IDE (ver. 20)
	Build automation Tool	ANT
Descrizione:	<p>Jumping Ball è un gioco Arcade multiplatforma. Il suo scopo consiste in una pallina salterina che deve evitare un granchio o un gabbiano oscillanti costantemente a destra e a manca. Inoltre, la pallina si trova al centro dello scenario e ci permane per tutta la durata del gioco. La difficoltà aumenta con il passare del tempo tramite il graduale aumento di velocità di movimento degli ostacoli. Ogni secondo di gioco si ottengono 20 punti in più. La pallina possiede inizialmente 3 vite e ne può guadagnare una ogni 2000 punti.</p>	

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Innanzitutto quando si avvia il gioco, si apre la finestra di Start in cui sono presenti 2 TextArea dove verranno poi trascritti i nomi, una volta finita la partita, dei giocatori e gli equivalenti punti ottenuti da ogni giocatore grazie allo scambio di dati ottenuto con la gestione dei file con estensione .csv. Non appena si preme il tasto Start si sente una delle nostre voci che dice che il gioco si sta avviando e grazie al metodo dispose() si chiude la finestra chiamata StartFrame e si apre la finestra MainFrame, ovvero la finestra di gioco.

```
java.awt.EventQueue.invokeLater(run() → {
    new MainFrame().setVisible(true);
    dispose();
});
} //GEN-LAST:event_startActionPerformed

private void helpActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_helpActionPerformed
    // TODO add your handling code here:

    // Eseguire il frame di aiuto
    java.awt.EventQueue.invokeLater(run() → { new HelpFrame().setVisible(true); });
} //GEN-LAST:event_helpActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)

    // Eseguire il frame iniziale
    java.awt.EventQueue.invokeLater(run() → { new StartFrame().setVisible(true); });
}

// Variables declaration - do not modify //GEN-BEGIN:variables
private javax.swing.JTextArea areaGiocatori;
private javax.swing.JTextArea areaPunteggi;
private javax.swing.JPanel classifica;
private javax.swing.JLabel etichettaGiocatori;
private javax.swing.JLabel etichettaPunteggi;
private javax.swing.JButton help;
private javax.swing.JScrollPane scorrimentoGiocatori;
private javax.swing.JScrollPane scorrimentoPunteggi;
private javax.swing.JLabel sfondoStart;
private javax.swing.JButton start;
// End of variables declaration //GEN-END:variables
```

Classe: 4B ITT.

a.s.: 2023/24.

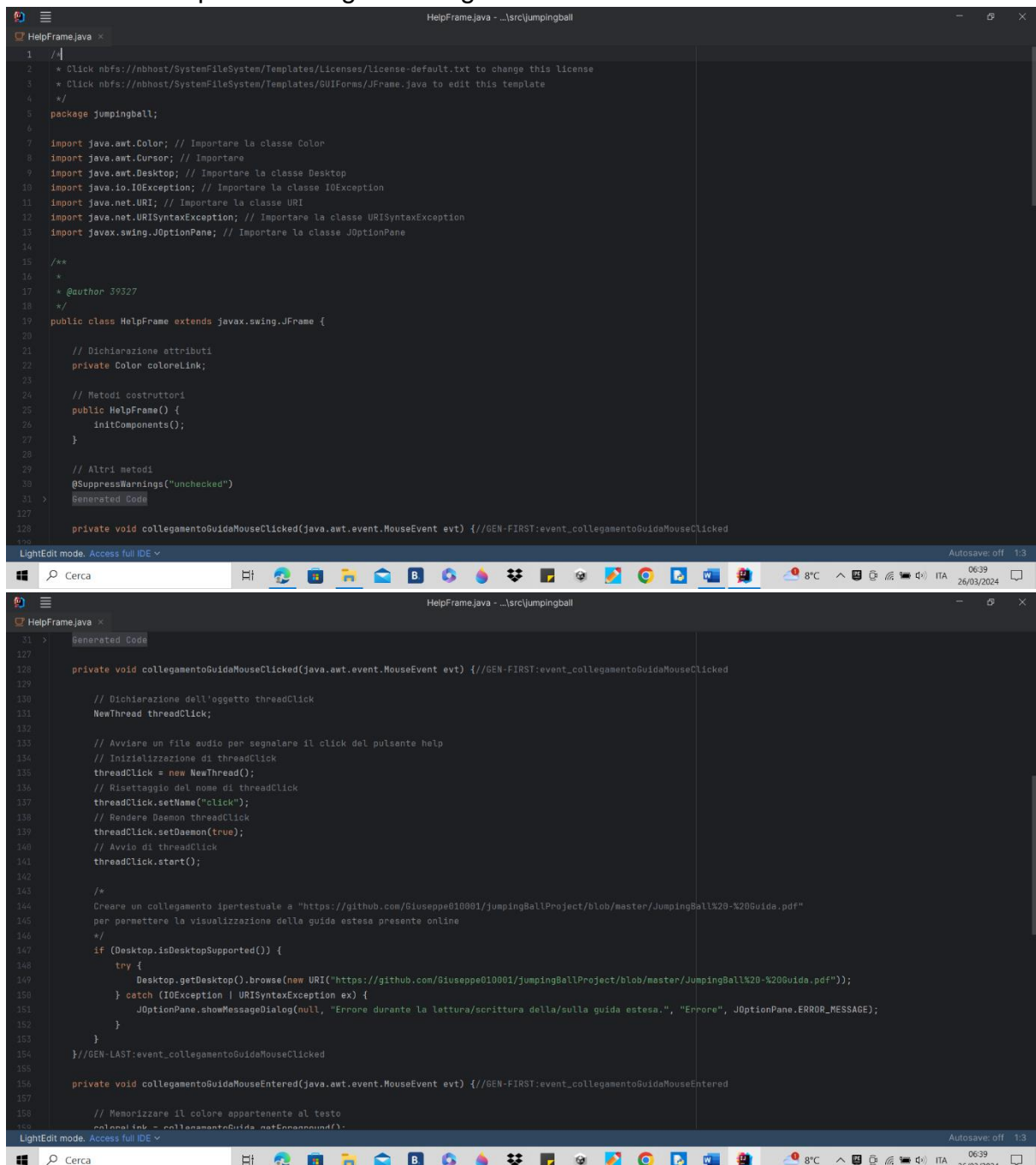
Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Classe HelpFrame:

Se si vuole sapere come funziona il nostro gioco basta semplicemente cliccare il punto interrogativo (in basso a destra) e si aprirà successivamente un'ulteriore finestra in cui è presente la guida del gioco.



```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/GuiForms/JFrame.java to edit this template
4  */
5  package jumpingball;
6
7  import java.awt.Color; // Importare la classe Color
8  import java.awt.Cursor; // Importare
9  import java.awt.Desktop; // Importare la classe Desktop
10 import java.io.IOException; // Importare la classe IOException
11 import java.net.URI; // Importare la classe URI
12 import java.net.URISyntaxException; // Importare la classe URISyntaxException
13 import javax.swing.JOptionPane; // Importare la classe JOptionPane
14
15 /**
16 *
17 * @author 39327
18 */
19 public class HelpFrame extends javax.swing.JFrame {
20
21     // Dichiarazione attributi
22     private Color coloreLink;
23
24     // Metodi costruttori
25     public HelpFrame() {
26         initComponents();
27     }
28
29     // Altri metodi
30     @SuppressWarnings("unchecked")
31     // Generated Code
32
33     private void collegamentoGuidaMouseClicked(java.awt.event.MouseEvent evt) {GEN-FIRST:event_collegamentoGuidaMouseClicked
34
35         // Dichiarazione dell'oggetto threadClick
36         NewThread threadClick;
37
38         // Avviare un file audio per segnalare il click del pulsante help
39         // Inizializzazione di threadClick
40         threadClick = new NewThread();
41         // Resettaggio del nome di threadClick
42         threadClick.setName("click");
43         // Rendere Daemon threadClick
44         threadClick.setDaemon(true);
45         // Avvio di threadClick
46         threadClick.start();
47
48         /*
49         Creare un collegamento ipertestuale a "https://github.com/Giuseppe010001/jumpingBallProject/blob/master/JumpingBall%20-%20Guida.pdf"
50         per permettere la visualizzazione della guida estesa presente online
51         */
52         if (Desktop.isDesktopSupported()) {
53             try {
54                 Desktop.getDesktop().browse(new URI("https://github.com/Giuseppe010001/jumpingBallProject/blob/master/JumpingBall%20-%20Guida.pdf"));
55             } catch (IOException | URISyntaxException ex) {
56                 JOptionPane.showMessageDialog(null, "Errore durante la lettura/scrittura della guida estesa.", "Errore", JOptionPane.ERROR_MESSAGE);
57             }
58         }
59     } //GEN-LAST:event_collegamentoGuidaMouseClicked
60
61     private void collegamentoGuidaMouseEntered(java.awt.event.MouseEvent evt) {GEN-FIRST:event_collegamentoGuidaMouseEntered
62
63         // Memorizzare il colore appartenente al testo
64         coloreLink = collegamentoGuida.getTextColor();
65     } //GEN-LAST:event_collegamentoGuidaMouseEntered
66 }
```

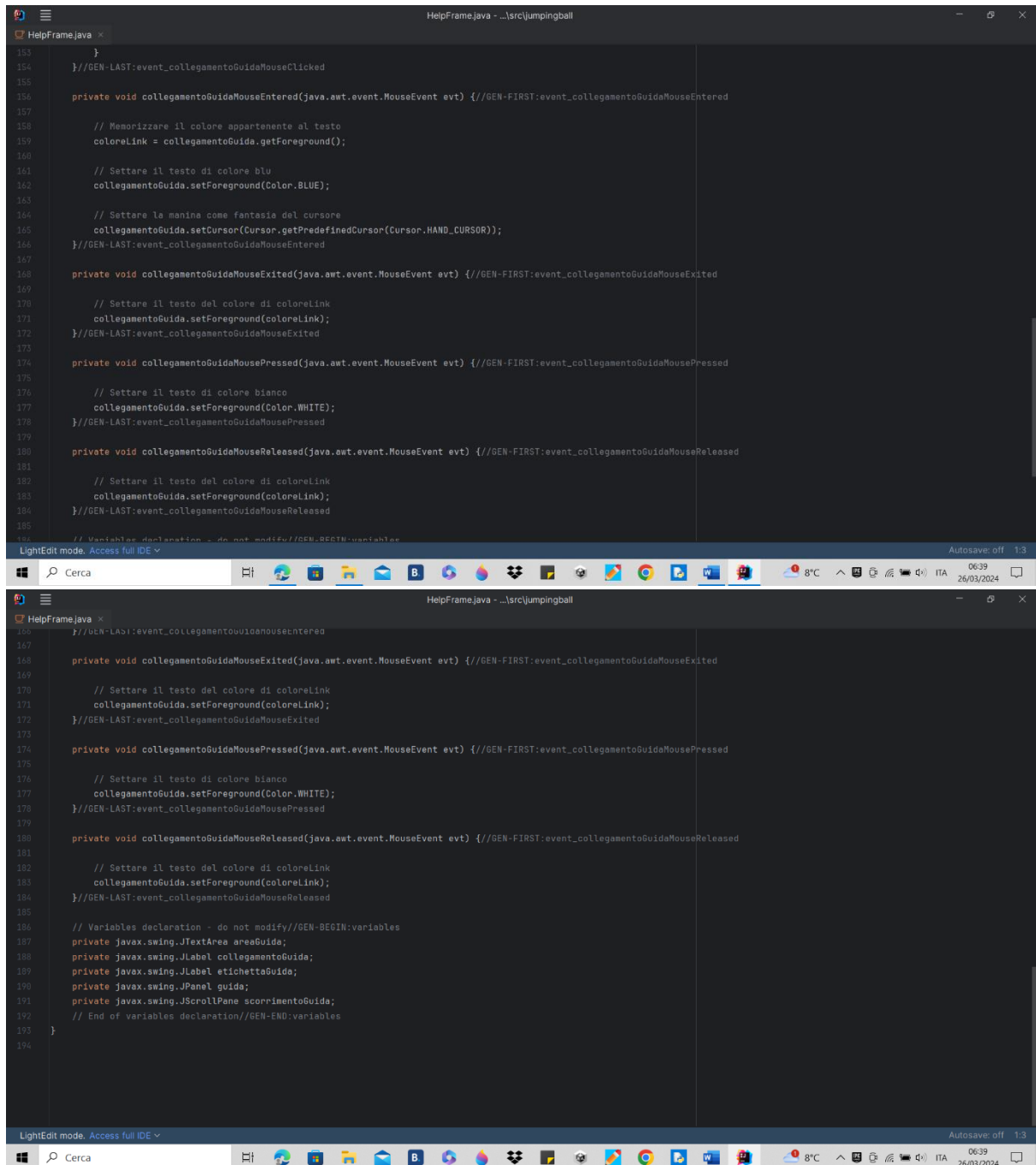
Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.



```
153 }
154 } //GEN-LAST:event_collegamentoGuidaMouseClicked
155
156 private void collegamentoGuidaMouseEntered(java.awt.event.MouseEvent evt) { //GEN-FIRST:event_collegamentoGuidaMouseEntered
157
158     // Memorizzare il colore appartenente al testo
159     coloreLink = collegamentoGuida.getForeground();
160
161     // Settare il testo di colore blu
162     collegamentoGuida.setForeground(Color.BLUE);
163
164     // Settare la manina come fantasia del cursore
165     collegamentoGuida.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
166 } //GEN-LAST:event_collegamentoGuidaMouseEntered
167
168 private void collegamentoGuidaMouseExited(java.awt.event.MouseEvent evt) { //GEN-FIRST:event_collegamentoGuidaMouseExited
169
170     // Settare il testo del colore di coloreLink
171     collegamentoGuida.setForeground(coloreLink);
172 } //GEN-LAST:event_collegamentoGuidaMouseExited
173
174 private void collegamentoGuidaMousePressed(java.awt.event.MouseEvent evt) { //GEN-FIRST:event_collegamentoGuidaMousePressed
175
176     // Settare il testo di colore bianco
177     collegamentoGuida.setForeground(Color.WHITE);
178 } //GEN-LAST:event_collegamentoGuidaMousePressed
179
180 private void collegamentoGuidaMouseReleased(java.awt.event.MouseEvent evt) { //GEN-FIRST:event_collegamentoGuidaMouseReleased
181
182     // Settare il testo del colore di coloreLink
183     collegamentoGuida.setForeground(coloreLink);
184 } //GEN-LAST:event_collegamentoGuidaMouseReleased
185
186 // Variables declaration - do not modify //GEN-BEGIN:variables
187 private javax.swing.JTextArea areaGuida;
188 private javax.swing.JLabel collegamentoGuida;
189 private javax.swing.JLabel etichettaGuida;
190 private javax.swing.JPanel guida;
191 private javax.swing.JScrollPane scorrimentoGuida;
192 // End of variables declaration //GEN-END:variables
193 }
194
```

```
167
168 private void collegamentoGuidaMouseExited(java.awt.event.MouseEvent evt) { //GEN-FIRST:event_collegamentoGuidaMouseExited
169
170     // Settare il testo del colore di coloreLink
171     collegamentoGuida.setForeground(coloreLink);
172 } //GEN-LAST:event_collegamentoGuidaMouseExited
173
174 private void collegamentoGuidaMousePressed(java.awt.event.MouseEvent evt) { //GEN-FIRST:event_collegamentoGuidaMousePressed
175
176     // Settare il testo di colore bianco
177     collegamentoGuida.setForeground(Color.WHITE);
178 } //GEN-LAST:event_collegamentoGuidaMousePressed
179
180 private void collegamentoGuidaMouseReleased(java.awt.event.MouseEvent evt) { //GEN-FIRST:event_collegamentoGuidaMouseReleased
181
182     // Settare il testo del colore di coloreLink
183     collegamentoGuida.setForeground(coloreLink);
184 } //GEN-LAST:event_collegamentoGuidaMouseReleased
185
186 // Variables declaration - do not modify //GEN-BEGIN:variables
187 private javax.swing.JTextArea areaGuida;
188 private javax.swing.JLabel collegamentoGuida;
189 private javax.swing.JLabel etichettaGuida;
190 private javax.swing.JPanel guida;
191 private javax.swing.JScrollPane scorrimentoGuida;
192 // End of variables declaration //GEN-END:variables
193 }
194
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Classe NewThread:

Mentre si gioca si può sentire una canzone delle 12 che abbiamo scelto, in modo randomico, e una volta finita ne viene riprodotta un'altra sempre in modo randomico, per far ciò abbiamo creato una sottoclasse NewThread della superclasse astratta Thread. Per far riprodurre ogni canzone abbiamo utilizzato il metodo run() tramite Overriding. Nella classe NewThread all'interno del primo blocco try...catch abbiamo usufruito di uno switch nel quale avviene l'assegnazione di un file audio a src Canzone scelto in modo randomico tra 12 canzoni già preimpostate, nel catch c'è l'eccezione nel caso di mancato supporto di un determinato formato audio. All'interno del secondo blocco try abbiamo usato vari metodi per la riproduzione delle canzoni, tra cui il metodo canzone.drain(); che permette di far eseguire tutta la canzone fino alla sua terminazione e l'eccezione del catch nel caso di assenza del file audio indicato. Infine nel terzo try viene calcolata la durata della canzone (in millisecondi), nel catch si verifica l'eccezione nel caso di errori nell'esecuzione di sleep, che all'interno del try mette in pausa il thread per un tempo pari a quello della canzone in questione. Alla fine della classe è stato implementato il finally grazie al quale si chiude il file utilizzato (il file della canzone) se la condizione del primo blocco è vera. Ecco le classi che abbiamo importato per far funzionare la classe:

```
package jumpingball;

import java.io.File; // Importare la classe File
import java.io.IOException; // Importare la classe IOException
import java.util.logging.Level; // Importare la classe Level
import java.util.logging.Logger; // Importare la classe Logger
import javax.sound.sampled.AudioFormat; // Importare la classe AudioFormat
import javax.sound.sampled.AudioInputStream; // Importare la classe AudioInputStream
import javax.sound.sampled.AudioSystem; // Importare la classe AudioSystem
import javax.sound.sampled.Clip; // Importare la classe Clip
import javax.sound.sampled.LineUnavailableException; // Importare la classe LineUnavailableException
import java.util.Random; // Importare la classe Random
import javax.sound.sampled.UnsupportedAudioFileException; // Importare la classe UnsupportedAudioFileException

/**
 *
 * @author 39327
 */
public class NewThread extends Thread {
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

```
try {

    // Assegnazione di un file audio a srcCanzone scelto in modo randomico tra 6 file preimpostati
    switch (genRand.nextInt(6)) {
        case 0 -> srcCanzone = AudioSystem.getAudioInputStream(new File("Baby Gang - Casablanca (feat. Morad) [Official Video].wav"));
        case 1 -> srcCanzone = AudioSystem.getAudioInputStream(new File("Baby Gang - Que Lo Ke [Official Lyric Video].wav"));
        case 2 -> srcCanzone = AudioSystem.getAudioInputStream(new File("Fred De Palma feat. Ana Mena - D'estate non vale (feat. Ana Mena).wav"));
        case 3 -> srcCanzone = AudioSystem.getAudioInputStream(new File("Giuni Russo - Unestate al mare.wav"));
        case 4 -> srcCanzone = AudioSystem.getAudioInputStream(new File("Il Pagante - @NewMusicItaly - Portofino - @NewMusicItaly.wav"));
        case 5 -> srcCanzone = AudioSystem.getAudioInputStream(new File("Kaoma - Lambada.wav"));
    }

    try {

        // Ottenimento del canale da utilizzare per l'esecuzione del file audio aperto
        canzone = AudioSystem.getClip();
        // Apertura del file assegnato a src_spostamento
        canzone.open(srcCanzone);
        // Inizio dell'esecuzione di tale file
        canzone.start();
        // Esecuzione di tale file fino alla sua terminazione
        canzone.drain();

        try {

            // Calcolare la durata di ascolto della canzone riprodotta in millisecondi
            formato = srcCanzone.getFormat();
            durataCanzone = (long) (canzone.getFrameLength() / formato.getFrameRate() * 1000);

            // Mettere in pausa il thread per un tempo pari a quello della canzone riprodotta
            sleep(durataCanzone);

            // Eccezione nel caso di errori nell'esecuzione di sleep
        } catch (InterruptedException ex) {
            Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
        }

        // Eccezione nel caso di assenza del file audio indicato
    } catch (LineUnavailableException ex) {
        Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

```
// Eccezione nel caso di mancato supporto di un determinato formato audio o nel caso di errore nello scambio di dati dal e al file audio utilizzato
} catch (UnsupportedAudioFileException | IOException ex) {
    Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
} finally {
    try {

        // Chiusura del file utilizzato
        srcCanzone.close();

        // Eccezione nel caso di errore nello scambio di dati dal e al file audio utilizzato
    } catch (IOException ex) {
        Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
    }
}

// Esecuzione ricorsiva del metodo run
run();
}
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Blocco di codice che permette l'incremento di 20 punti ogni secondo e l'incremento delle vite ogni 2000 punti, inoltre ogni volta che si guadagna una vita si sente in sottofondo una delle nostre voci che sta ad indicare l'incremento di una vita oppure il decremento di vite.

```
NewThread.java ×

// Esecuzione ricorsiva del metodo run
run();
} case "incrementoPunti" -> {

// Mostrare graficamente il punteggio in tempo reale
frameP.getPunteggio().setText("Punti: " + (frameP.getNPunti()));

// Incrementare il numero di vite ad intervalli di 2000 punti
if (frameP.getNPunti() != 0 && frameP.getNPunti() % 2000 == 0) {

// Incrementare il numero di vite
frameP.incrementoNVite();
frameP.getVite().setText("Vite: " + (frameP.getNVite()));

// Fare in modo che al prossimo richiamo di run venga eseguito un altro blocco di codice dello switch
this.setName("incrementoVita");
} else {
try {

// Mettere in pausa il thread per un tempo pari a un secondo
sleep(1000);

// Eccezione nel caso di errori nell'esecuzione di sleep
} catch (InterruptedException ex) {
Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
}
}

// Incrementare il numero di punti
frameP.incrementoNPunti();
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Questo blocco di codice permette la riproduzione del “sii” quando il giocatore guadagna una vita.

```
    } case "incrementoVita" -> {  
  
        // Dichiarazione degli oggetti srcIncremento, formatoIncremento, e incremento appartenenti rispettivamente alle d  
        AudioInputStream srcIncremento = null;  
        AudioFormat formatoIncremento;  
        Clip incremento;  
  
        // Dichiarazione variabili  
        long durataIncremento;  
  
        try {  
  
            // Assegnazione del file audio "incrementoVita.wav" a srcIncremento  
            srcIncremento = AudioSystem.getAudioInputStream(new File("incrementoVita.wav"));  
  
            try {  
  
                // Ottenimento del canale da utilizzare per l'esecuzione del file audio aperto  
                incremento = AudioSystem.getClip();  
                // Apertura del file assegnato a srcSalto  
                incremento.open(srcIncremento);  
                // Inizio dell'esecuzione di tale file  
                incremento.start();  
                // Esecuzione di tale file fino alla sua terminazione  
                incremento.drain();  
  
                try {  
  
                    // Calcolare la durata di ascolto del file riprodotto in millisecondi  
                    formatoIncremento = srcIncremento.getFormat();  
                    durataIncremento = (long) (incremento.getFrameLength() / formatoIncremento.getFrameRate() * 1000);  

```


Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

```
try {

    // Assegnazione del file audio "incrementoVita.wav" a srcIncremento
    srcIncremento = AudioSystem.getAudioInputStream(new File("incrementoVita.wav"));

    try {

        // Ottenimento del canale da utilizzare per l'esecuzione del file audio aperto
        incremento = AudioSystem.getClip();
        // Apertura del file assegnato a srcSalto
        incremento.open(srcIncremento);
        // Inizio dell'esecuzione di tale file
        incremento.start();
        // Esecuzione di tale file fino alla sua terminazione
        incremento.drain();

        try {

            // Calcolare la durata di ascolto del file riprodotto in millisecondi
            formatoIncremento = srcIncremento.getFormat();
            durataIncremento = (long) (incremento.getFrameLength() / formatoIncremento.getFrameRate() * 1000);

            // Mettere in pausa il thread per un tempo pari a quello del file audio riprodotto
            sleep(durataIncremento);

            // Eccezione nel caso di errori nell'esecuzione di sleep
        } catch (InterruptedException ex) {
            Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

```
// Eccezione nel caso di assenza del file audio indicato
} catch (FileNotFoundException ex) {
    Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
}

// Eccezione nel caso di mancato supporto di un determinato formato audio o nel caso di errore nello scambio di dati dal e al file audio utilizzato
} catch (UnsupportedAudioFormatException | IOException ex) {
    Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
} finally {
    try {

        // Chiusura del file utilizzato
        srcIncremento.close();

        // Eccezione nel caso di errore nello scambio di dati dal e al file audio utilizzato
    } catch (IOException ex) {
        Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
    }
}

if (frameP.getVelocitaMovimento() > 5)
    frameP.decrementoVelocitaMovimento();

this.setName("incrementoPunti");
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Questo blocco di codice verifica tutte le volte il conflitto tra granchio o gabbiano e la pallina se questo non avviene mette in pausa il thread per un tempo pari al valore ritornato da getVelocitaMovimento.

```
} case "conflitto" -> {
    while (frameP.getNVite() > 0) {

        if ((frameP.getGranchio().getX() >= frameP.getPallina().getX() - 32 && frameP.getGranchio().getX() <= frameP.getPallina().getX() + 32 && frameP.getGranchio().getY() == frameP.getPallina().getY() ||
            (frameP.getGabbiano().getX() >= frameP.getPallina().getX() - 32 && frameP.getGabbiano().getX() <= frameP.getPallina().getX() + 32 && frameP.getGabbiano().getY() == frameP.getPallina().getY())) {
            this.setName("decrementoVita");
            run();
        } else {
            try {
                sleep(frameP.getVelocitaMovimento());
            } catch (InterruptedException ex) {
                Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }

    frameP.getPallina().setVisible(false);
    this.setName("musica");
    run();
} case "decrementoVita" -> {

    // Dichiarazione degli oggetti srcDecremento, formatoDecremento, e decremento appartenenti rispettivamente alle classi AudioInputStream, AudioFormat, e Clip e implementazioni
    AudioInputStream srcDecremento = null;
    AudioFormat formatoDecremento;
    Clip decremento;

    // Dichiarazione variabili
    long durataDecremento;

    // Decrementare il numero di vite
    frameP.decrementoNVite();
    frameP.getVite().setText("Vite: " + (frameP.getNVite()));
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Qua il codice non fa altro che gestire il decremento delle vite e quindi gestisce la riproduzione dell'audio di uno di noi 3 che riproduce un “noo” quando il giocatore perde una vita.

```
} case "decrementoVita" -> {

    // Dichiarazione degli oggetti srcDecremento, formatoDecremento, e decremento appartenenti rispettivamente a InputStream, AudioFormat, e Clip
    AudioInputStream srcDecremento = null;
    AudioFormat formatoDecremento;
    Clip decremento;

    // Dichiarazione variabili
    long durataDecremento;

    // Decrementare il numero di vite
    frameP.decrementoNVite();
    frameP.getVite().setText("Vite: " + (frameP.getNVite()));

    try {

        // Assegnazione del file audio "decrementoVita.wav" a srcDecremento
        srcDecremento = AudioSystem.getAudioInputStream(new File("decrementoVita.wav"));

        try {

            // Ottenimento del canale da utilizzare per l'esecuzione del file audio aperto
            decremento = AudioSystem.getClip();
            // Apertura del file assegnato a srcDecremento
            decremento.open(srcDecremento);
            // Inizio dell'esecuzione di tale file
            decremento.start();
            // Esecuzione di tale file fino alla sua terminazione
            decremento.drain();

            try {

                // Calcolare la durata di ascolto del file riprodotto in millisecondi
                formatoDecremento = srcDecremento.getFormat();
                durataDecremento = (Long) (decremento.getFrameLength() / formatoDecremento.getFrameRate() * 1000);

                // Mettere in pausa il thread per un tempo pari a quello del file audio riprodotto
                sleep(durataDecremento);

                // Eccezione nel caso di errori nell'esecuzione di sleep
            } catch (InterruptedException ex) {
                Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
            }

            // Eccezione nel caso di assenza del file audio indicato
            } catch (LineUnavailableException ex) {
                Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
            }

            // Eccezione nel caso di mancato supporto di un determinato formato audio o nel caso di errore nello scambio di dati dal e al file audio utilizzato
        } catch (UnsupportedAudioFileException | IOException ex) {
            Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
        } finally {
            try {

                // Chiusura del file utilizzato
                srcDecremento.close();
            }
        }
    }
}
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Qua invece abbiamo 2 cicli while() in cui il primo permette il salto della pallina verso l'alto e il secondo while permette il "ritorno" della palla verso il suolo.

```
} case "pallina" -> {
    int G = 4;

    while (frameP.getYPallina() >= 216) {
        try {
            sleep(frameP.getVelocitaMovimento());
        } catch (InterruptedException ex) {
            Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
        }

        frameP.getPallina().setLocation(frameP.getPallina().getX(), frameP.getYPallina());

        if ((280 - frameP.getYPallina()) % 16 == 0)
            G--;

        frameP.decrementoYPallina(G);
    }

    G = 1;
```

```
G = 1;

while (frameP.getYPallina() <= 280) {
    try {
        sleep(frameP.getVelocitaMovimento());
    } catch (InterruptedException ex) {
        Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
    }

    frameP.getPallina().setLocation(frameP.getPallina().getX(), frameP.getYPallina());

    if ((frameP.getYPallina() - 216) % 16 == 0)
        G++;

    frameP.incrementoYPallina(G);
}

frameP.decrementoYPallina(G);
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Questo blocco codice che ci permette di far saltare la nostra pallina e ogni volta che salta si sentirà un “boing” detto da uno di noi:

```
} case "salto" -> {

    // Dichiarazione degli oggetti srcSalto, formato, e salto appartenenti rispettivamente alle classi
    AudioInputStream srcSalto = null;
    AudioFormat formatoSalto;
    Clip salto;

    // Dichiarazione variabili
    long durataSalto;

    try {

        // Assegnazione del file audio "salto.wav" a srcSalto
        srcSalto = AudioSystem.getAudioInputStream(new File("salto.wav"));

        try {

            // Ottenimento del canale da utilizzare per l'esecuzione del file audio aperto
            salto = AudioSystem.getClip();
            // Apertura del file assegnato a srcSalto
            salto.open(srcSalto);
            // Inizio dell'esecuzione di tale file
            salto.start();
            // Esecuzione di tale file fino alla sua terminazione
            salto.drain();

            try {

                // Calcolare la durata di ascolto del file riprodotto in millisecondi
                formatoSalto = srcSalto.getFormat();
                durataSalto = (long) (salto.getFrameLength() / formatoSalto.getFrameRate() * 1000);

                sleep(durataSalto);

                // Eccezione nel caso di errori nell'esecuzione di sleep
            } catch (InterruptedException ex) {
                Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
            }

            // Eccezione nel caso di assenza del file audio indicato
        } catch (LineUnavailableException ex) {
            Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
        }

        // Eccezione nel caso di mancato supporto di un determinato formato audio o nel caso di errore nello scambio di dati dal e al file audio utilizzato
    } catch (UnsupportedAudioFormatException | IOException ex) {
        Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
    } finally {
        try {

            // Chiusura del file utilizzato
            srcSalto.close();

            // Eccezione nel caso di errore nello scambio di dati dal e al file audio utilizzato
        } catch (IOException ex) {
            Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Ecco come abbiamo implementato la classe NewThread in modo da poter generare in modo randomico lo scorrimento del gabbiano e del granchio da sinistra verso destra e viceversa.

```
    } case "granchio" -> {
        Random genRand = new Random();

        switch (genRand.nextInt(2)) {
            case 0 -> {
                while (true) {
                    if (frameP.getX0stacoli() >= 600) {
                        switch (genRand.nextInt(2)) {
                            case 0 -> {
                                frameP.ripristinoX0stacoliAvanti();
                                frameP.getGranchio().setLocation(frameP.getX0stacoli(), frameP.getGranchio().getY());
                            } case 1 -> {
                                this.setName("gabbiano");
                                run();
                            }
                        }
                    }
                }

                frameP.incrementoX0stacoli();

                frameP.getGranchio().setLocation(frameP.getX0stacoli(), frameP.getGranchio().getY());

                try {
                    sleep(frameP.getVelocitaMovimento());
                } catch (InterruptedException ex) {
                    Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
                }
            }
        }
    }
```

```
    } case 1 -> {
        while (true) {
            if (frameP.getX0stacoli() <= -64) {
                switch (genRand.nextInt(2)) {
                    case 0 -> {
                        frameP.ripristinoX0stacoliIndietro();
                        frameP.getGranchio().setLocation(frameP.getX0stacoli(), frameP.getGranchio().getY());
                    } case 1 -> {
                        this.setName("gabbiano");
                        run();
                    }
                }
            }
        }

        frameP.decrementoX0stacoli();

        frameP.getGranchio().setLocation(frameP.getX0stacoli(), frameP.getGranchio().getY());

        try {
            sleep(frameP.getVelocitaMovimento());
        } catch (InterruptedException ex) {
            Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

```
    }
} case "gabbiano" -> {
    Random genRand = new Random();

    switch (genRand.nextInt(2)) {
        case 0 -> {
            while (true) {
                if (frameP.getX0stacoli() >= 600) {
                    switch (genRand.nextInt(2)) {
                        case 0 -> {
                            frameP.ripristinoX0stacoliAvanti();
                            frameP.getGabbiano().setLocation(frameP.getX0stacoli(), frameP.getGabbiano().getY());
                        } case 1 -> {
                            this.setName("granchio");
                            run();
                        }
                    }
                }
            }

            frameP.incrementoX0stacoli();

            frameP.getGabbiano().setLocation(frameP.getX0stacoli(), frameP.getGabbiano().getY());

            try {
                sleep(frameP.getVelocitaMovimento());
            } catch (InterruptedException ex) {
                Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
}
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

In questo blocco di gioco l'if vale a dire che il gabbiano scorre nello sfondo di gioco orizzontalmente fino alla posizione (600;216) e poi riprende o da (-64;216) scorrendo da sinistra verso destra oppure riprende da (600;216) scorrendo da destra verso sinistra.

```
    } case 1 -> {
        while (true) {
            if (frameP.getX0stacoli() <= -64) {
                switch (genRand.nextInt(2)) {
                    case 0 -> {
                        frameP.ripristinoX0stacoliIndietro();
                        frameP.getGabbiano().setLocation(frameP.getX0stacoli(), frameP.getGabbiano().getY());
                    } case 1 -> {
                        this.setName("granchio");
                        run();
                    }
                }
            }
        }

        frameP.decrementoX0stacoli();

        frameP.getGabbiano().setLocation(frameP.getX0stacoli(), frameP.getGabbiano().getY());

        try {
            sleep(frameP.getVelocitaMovimento());
        } catch (InterruptedException ex) {
            Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```


Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Questa parte di codice che ti permette di riprodurre un file audio che prova a simulare il click di un pulsante mediante uno schiocco di dita.

```
} case "click" -> {

    // Dichiarazione degli oggetti srcClick, formatoClick, e click appartenenti rispettivamente alle classi
    AudioInputStream srcClick = null;
    AudioFormat formatoClick;
    Clip click;

    // Dichiarazione variabili
    long durataClick;

    try {

        // Assegnazione del file audio "click.wav" a srcClick
        srcClick = AudioSystem.getAudioInputStream(new File("click.wav"));

        try {

            // Ottenimento del canale da utilizzare per l'esecuzione del file audio aperto
            click = AudioSystem.getClip();
            // Apertura del file assegnato a srcClick
            click.open(srcClick);
            // Inizio dell'esecuzione di tale file
            click.start();
            // Esecuzione di tale file fino alla sua terminazione
            click.drain();

            try {

                // Calcolare la durata di ascolto del file riprodotto in millisecondi
                formatoClick = srcClick.getFormat();
                durataClick = (long) (click.getFrameLength() / formatoClick.getFrameRate() * 1000);

                durataClick = (long) (click.getFrameLength() / formatoClick.getFrameRate() * 1000);

                // Mettere in pausa il thread per un tempo pari a quello del file audio riprodotto
                sleep(durataClick);

                // Eccezione nel caso di errori nell'esecuzione di sleep
            } catch (InterruptedException ex) {
                Logger.getLogger(NewThread.class.getName()).log(Level.SEVERE, null, ex);
            }

            // Eccezione nel caso di assenza del file audio indicato
            } catch (LineUnavailableException ex) {
                Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
            }

            // Eccezione nel caso di mancato supporto di un determinato formato audio o nel caso di errore nello scambio di dati da
        } catch (UnsupportedAudioFileException | IOException ex) {
            Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
        } finally {
            try {

                // Chiusura del file utilizzato
                srcClick.close();

                // Eccezione nel caso di errore nello scambio di dati dal e al file audio utilizzato
            } catch (IOException ex) {
                Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
}
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Classe MainFrame:

Ecco come abbiamo gestito l'evento del click della pallina nella classe MainFrame. Tale click permette di richiamare 2 thread che parallelamente eseguono l'azione del salto e l'effetto sonoro relativo di quest'ultimo.

```
private void pallinaMouseClicked(java.awt.event.MouseEvent evt) {GEN-FIRST:event_pallinaMouseClicked
    if (pallina.getY() == 280) {
        NewThread threadPallina, threadSalto;

        threadPallina = new NewThread(this);
        threadPallina.setName("pallina");
        threadPallina.start();

        threadSalto = new NewThread();
        threadSalto.setName("salto");
        threadSalto.start();
    }
}
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Inoltre, nel MainFrame, è stato implementato il blocco di codice per la gestione del file Classifica.csv sia per la lettura sia per la scrittura:

Per la lettura, viene seguito il while finché ci sono righe, e quindi dei nomi e dei punteggi, dividiamo ogni campo del record da un ';' ed estraiamo ogni elemento in una variabile opportuna, sia per il nome del giocatore sia per il punteggio ottenuto da quel giocatore;

```
try {
    BufferedReader Lettore = new BufferedReader(new FileReader(nomeFile));
    //readLine() metodo della classe che consent di leggere una riga
    Lettore.readLine();
    while((riga = Lettore.readLine())!=null)
    {
        statoGiocatore = riga.split(";");
        //estrae ogni elemento in una variabile opportuna
        nomeGiocatore = statoGiocatore[0];
        punteggio = statoGiocatore[1];

        frameInizio.getAreaGiocatori().append(nomeGiocatore + '\n');
        frameInizio.getAreaPunteggi().append(punteggio + '\n');
    }
} catch (FileNotFoundException ex) {
    System.out.println("Impossibile trovare il file " + nomeFile);
} catch (IOException ex) {
    System.out.println("Errore nella lettura del file " + nomeFile);
}
```

a.s.: 2023/24.

Nome del gruppo: Galaxy Java Development Team.

Per la scrittura, quest'ultima avviene solo se vi è la necessità di aggiornare il contenuto della classifica, che si fa prima mediante la sovrascrittura di quanto presente in `Classifica.csv` e dopo con il resettaggio del testo della classifica nello `StartFrame`. L'eventuale scrittura avviene dopo la pressione del tasto `Confirm` nel `MainFrame`.

```
288 private void confirmActionPerformed(java.awt.event.ActionEvent evt) {GEN-FIRST:event_confirmActionPerformed
289
290     // Dichiarazione (e implementazione) variabili
291     String nomeGiocatoreLetto, punteggioLetto, nomeGiocatoreCambio, punteggioCambio, costruttoreRiga = "", riga, nomeFile = "Classifica.csv";
292
293     // Dichiarazione array
294     String[] statoGiocatore;
295
296     /*
297     Dichiarazione (e implementazione) degli oggetti frameInizio, scrittura e lettura
298     */
299     StartFrame frameInizio = new StartFrame();
300     PrintWriter scrittura = null;
301     BufferedReader lettura;
302
303     try {
304
305         // Inizializzare l'oggetto Lettore
306         lettura = new BufferedReader(new FileReader(nomeFile));
307
308         // Aggiornare la classifica con la solita prima riga di testo indicativa dei dati che si vanno a leggere
309         costruttoreRiga += "Giocatore;Punteggio\n";
310
311         // Saltare la prima riga del file "Classifica.csv"
312         lettura.readLine();
313
314         // Lettura della seconda riga del file "Classifica.csv"
315         riga = lettura.readLine();
316
317         // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
318         statoGiocatore = riga.split(";");
319
320         // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
321
322     } catch (IOException ex) {
323         Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
324     }
325
326     // Aggiornare la classifica con la solita prima riga di testo indicativa dei dati che si vanno a leggere
327     costruttoreRiga += "Giocatore;Punteggio\n";
328
329     // Saltare la prima riga del file "Classifica.csv"
330     lettura.readLine();
331
332     // Lettura della seconda riga del file "Classifica.csv"
333     riga = lettura.readLine();
334
335     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
336     statoGiocatore = riga.split(";");
337
338     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
339     nomeGiocatoreLetto = statoGiocatore[0];
340
341     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
342     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
343
344     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
345     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
346
347         // Aggiungere il prossimo risultato alla classifica aggiornata
348         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
349
350         // Leggere il contenuto della prossima riga della vecchia classifica
351         riga = lettura.readLine();
352     }
353
354     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
355     statoGiocatore = riga.split(";");
356
357     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
358     nomeGiocatoreLetto = statoGiocatore[0];
359
360     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
361     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
362
363     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
364     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
365
366         // Aggiungere il prossimo risultato alla classifica aggiornata
367         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
368
369         // Leggere il contenuto della prossima riga della vecchia classifica
370         riga = lettura.readLine();
371     }
372
373     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
374     statoGiocatore = riga.split(";");
375
376     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
377     nomeGiocatoreLetto = statoGiocatore[0];
378
379     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
380     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
381
382     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
383     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
384
385         // Aggiungere il prossimo risultato alla classifica aggiornata
386         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
387
388         // Leggere il contenuto della prossima riga della vecchia classifica
389         riga = lettura.readLine();
390     }
391
392     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
393     statoGiocatore = riga.split(";");
394
395     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
396     nomeGiocatoreLetto = statoGiocatore[0];
397
398     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
399     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
400
401     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
402     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
403
404         // Aggiungere il prossimo risultato alla classifica aggiornata
405         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
406
407         // Leggere il contenuto della prossima riga della vecchia classifica
408         riga = lettura.readLine();
409     }
410
411     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
412     statoGiocatore = riga.split(";");
413
414     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
415     nomeGiocatoreLetto = statoGiocatore[0];
416
417     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
418     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
419
420     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
421     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
422
423         // Aggiungere il prossimo risultato alla classifica aggiornata
424         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
425
426         // Leggere il contenuto della prossima riga della vecchia classifica
427         riga = lettura.readLine();
428     }
429
430     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
431     statoGiocatore = riga.split(";");
432
433     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
434     nomeGiocatoreLetto = statoGiocatore[0];
435
436     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
437     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
438
439     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
440     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
441
442         // Aggiungere il prossimo risultato alla classifica aggiornata
443         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
444
445         // Leggere il contenuto della prossima riga della vecchia classifica
446         riga = lettura.readLine();
447     }
448
449     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
450     statoGiocatore = riga.split(";");
451
452     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
453     nomeGiocatoreLetto = statoGiocatore[0];
454
455     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
456     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
457
458     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
459     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
460
461         // Aggiungere il prossimo risultato alla classifica aggiornata
462         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
463
464         // Leggere il contenuto della prossima riga della vecchia classifica
465         riga = lettura.readLine();
466     }
467
468     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
469     statoGiocatore = riga.split(";");
470
471     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
472     nomeGiocatoreLetto = statoGiocatore[0];
473
474     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
475     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
476
477     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
478     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
479
480         // Aggiungere il prossimo risultato alla classifica aggiornata
481         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
482
483         // Leggere il contenuto della prossima riga della vecchia classifica
484         riga = lettura.readLine();
485     }
486
487     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
488     statoGiocatore = riga.split(";");
489
490     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
491     nomeGiocatoreLetto = statoGiocatore[0];
492
493     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
494     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
495
496     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
497     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
498
499         // Aggiungere il prossimo risultato alla classifica aggiornata
500         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
501
502         // Leggere il contenuto della prossima riga della vecchia classifica
503         riga = lettura.readLine();
504     }
505
506     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
507     statoGiocatore = riga.split(";");
508
509     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
510     nomeGiocatoreLetto = statoGiocatore[0];
511
512     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
513     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
514
515     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
516     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
517
518         // Aggiungere il prossimo risultato alla classifica aggiornata
519         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
520
521         // Leggere il contenuto della prossima riga della vecchia classifica
522         riga = lettura.readLine();
523     }
524
525     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
526     statoGiocatore = riga.split(";");
527
528     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
529     nomeGiocatoreLetto = statoGiocatore[0];
530
531     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
532     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
533
534     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
535     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
536
537         // Aggiungere il prossimo risultato alla classifica aggiornata
538         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
539
540         // Leggere il contenuto della prossima riga della vecchia classifica
541         riga = lettura.readLine();
542     }
543
544     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
545     statoGiocatore = riga.split(";");
546
547     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
548     nomeGiocatoreLetto = statoGiocatore[0];
549
550     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
551     punteggioLetto = Integer.parseInt(statoGiocatore[1]);
552
553     // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
554     while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
555
556         // Aggiungere il prossimo risultato alla classifica aggiornata
557         costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
558
559         // Leggere il contenuto della prossima riga della vecchia classifica
560         riga = lettura.readLine();
561     }
562
563     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
564     statoGioc
```

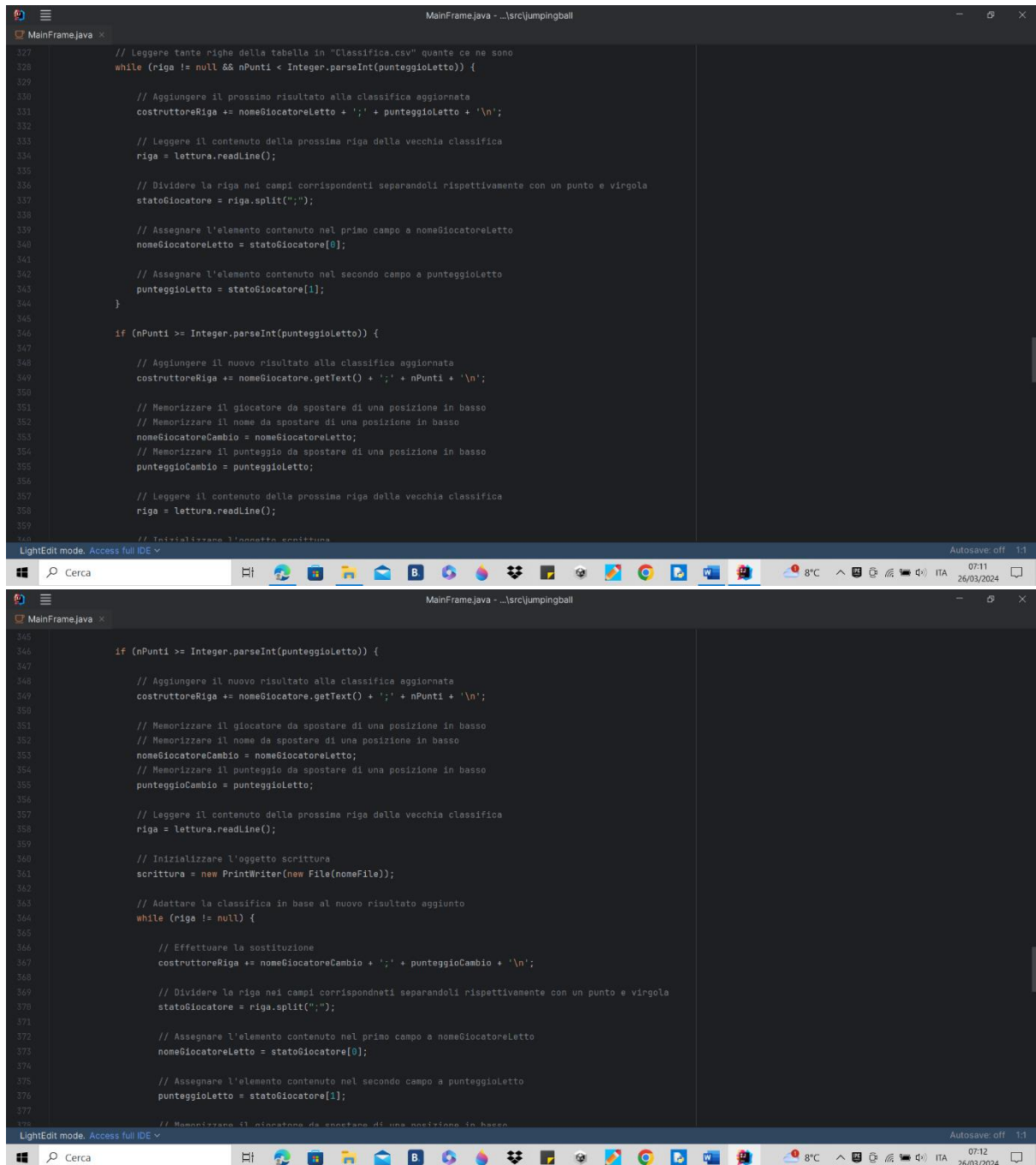
Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.



```
327 // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono
328 while (riga != null && nPunti < Integer.parseInt(punteggioLetto)) {
329
330     // Aggiungere il prossimo risultato alla classifica aggiornata
331     costruttoreRiga += nomeGiocatoreLetto + ';' + punteggioLetto + '\n';
332
333     // Leggere il contenuto della prossima riga della vecchia classifica
334     riga = lettura.readLine();
335
336     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
337     statoGiocatore = riga.split(";");
338
339     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
340     nomeGiocatoreLetto = statoGiocatore[0];
341
342     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
343     punteggioLetto = statoGiocatore[1];
344 }
345
346 if (nPunti >= Integer.parseInt(punteggioLetto)) {
347
348     // Aggiungere il nuovo risultato alla classifica aggiornata
349     costruttoreRiga += nomeGiocatore.getText() + ';' + nPunti + '\n';
350
351     // Memorizzare il giocatore da spostare di una posizione in basso
352     // Memorizzare il nome da spostare di una posizione in basso
353     nomeGiocatoreCambio = nomeGiocatoreLetto;
354     // Memorizzare il punteggio da spostare di una posizione in basso
355     punteggioCambio = punteggioLetto;
356
357     // Leggere il contenuto della prossima riga della vecchia classifica
358     riga = lettura.readLine();
359
360     // Inizializzare l'oggetto scrittura
361     scrittura = new PrintWriter(new File(nomeFile));
362
363     // Adattare la classifica in base al nuovo risultato aggiunto
364     while (riga != null) {
365
366         // Effettuare la sostituzione
367         costruttoreRiga += nomeGiocatoreCambio + ';' + punteggioCambio + '\n';
368
369         // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
370         statoGiocatore = riga.split(";");
371
372         // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
373         nomeGiocatoreLetto = statoGiocatore[0];
374
375         // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
376         punteggioLetto = statoGiocatore[1];
377
378         // Memorizzare il giocatore da spostare di una posizione in basso
```

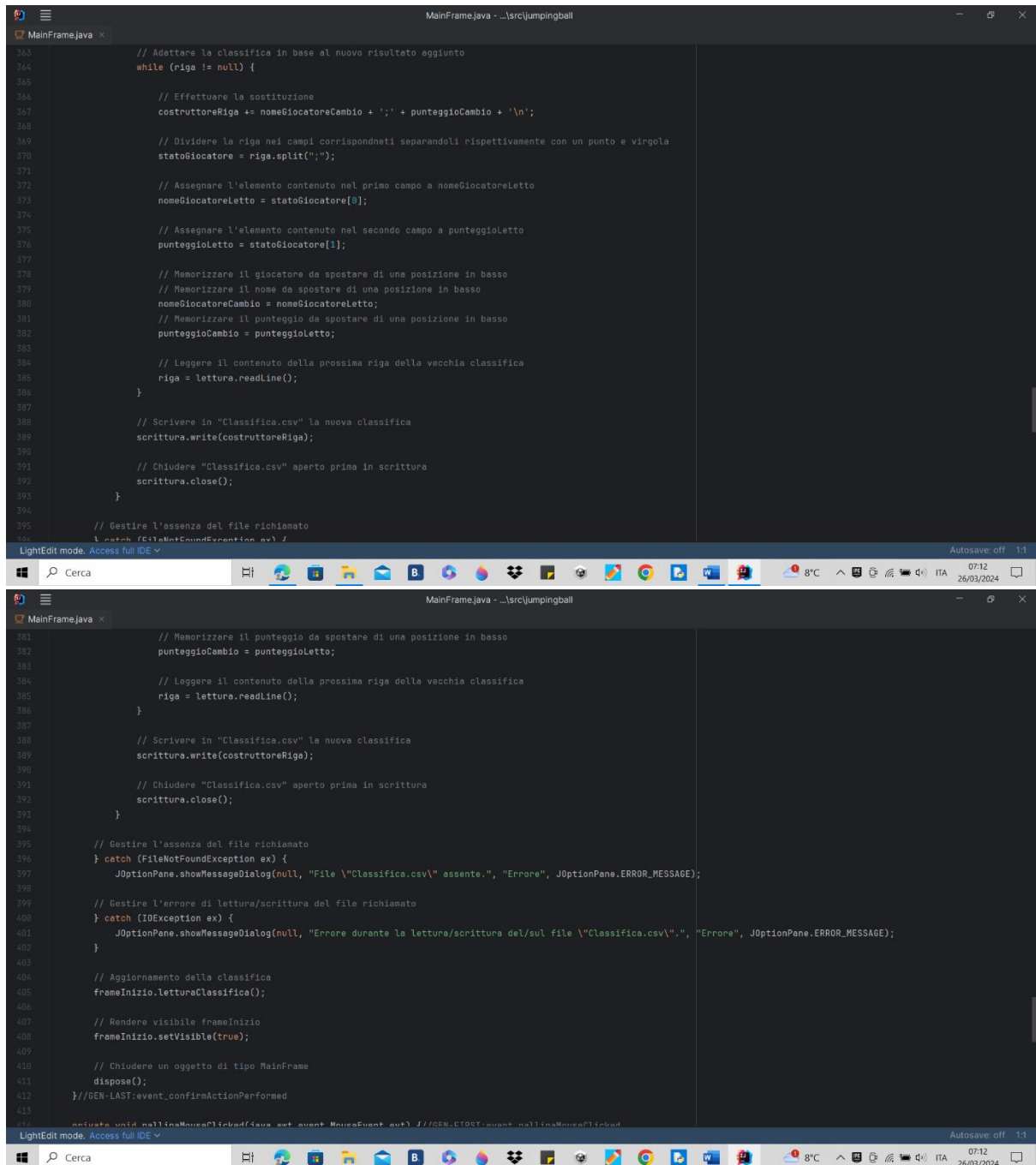
Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.



```
363 // Adattare la classifica in base al nuovo risultato aggiunto
364 while (riga != null) {
365
366     // Effettuare la sostituzione
367     costruttoreRiga += nomeGiocatoreCambio + ';' + punteggioCambio + '\n';
368
369     // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola
370     statoGiocatore = riga.split(";");
371
372     // Assegnare l'elemento contenuto nel primo campo a nomeGiocatoreLetto
373     nomeGiocatoreLetto = statoGiocatore[0];
374
375     // Assegnare l'elemento contenuto nel secondo campo a punteggioLetto
376     punteggioLetto = statoGiocatore[1];
377
378     // Memorizzare il giocatore da spostare di una posizione in basso
379     // Memorizzare il nome da spostare di una posizione in basso
380     nomeGiocatoreCambio = nomeGiocatoreLetto;
381     // Memorizzare il punteggio da spostare di una posizione in basso
382     punteggioCambio = punteggioLetto;
383
384     // Leggere il contenuto della prossima riga della vecchia classifica
385     riga = lettura.readLine();
386 }
387
388 // Scrivere in "Classifica.csv" la nuova classifica
389 scrittura.write(costruttoreRiga);
390
391 // Chiudere "Classifica.csv" aperto prima in scrittura
392 scrittura.close();
393 }
394
395 // Gestire l'assenza del file richiamato
396 } catch (FileNotFoundException ex) {
397
398     // Memorizzare il punteggio da spostare di una posizione in basso
399     punteggioCambio = punteggioLetto;
400
401     // Leggere il contenuto della prossima riga della vecchia classifica
402     riga = lettura.readLine();
403 }
404
405 // Scrivere in "Classifica.csv" la nuova classifica
406 scrittura.write(costruttoreRiga);
407
408 // Chiudere "Classifica.csv" aperto prima in scrittura
409 scrittura.close();
410 }
411
412 // Gestire l'errore di lettura/scrittura del file richiamato
413 } catch (IOException ex) {
414     JOptionPane.showMessageDialog(null, "Errore durante la lettura/scrittura del/sul file \"Classifica.csv\".", "Errore", JOptionPane.ERROR_MESSAGE);
415 }
416
417 // Aggiornamento della classifica
418 frameInizio.letturaClassifica();
419
420 // Rendere visibile frameInizio
421 frameInizio.setVisible(true);
422
423 // Chiudere un oggetto di tipo JFrame
424 dispose();
425 } //GEN-LAST:event_confirmActionPerformed
426
427 // Gestire il click del pulsante void nallinaMouseClicked(java.awt.event.MouseEvent evt) //GEN-LAST:event_nallinaMouseClicked
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Classe StartFrame:

Nello StartFrame in basso a destra si può notare un punto interrogativo che se viene cliccato si apre una nuova finestra con su scritta tutta la guida del gioco e la si può leggere muovendo il cursore che si trova a destra verso il basso.

```
private void helpActionPerformed(java.awt.event.ActionEvent evt) {GEN-FIRST:event_helpActionPerformed
|
    // Dichiarazione e inizializzazione dell'oggetto frameAiuto della classe HelpFrame e sola dichiarazione dell'oggetto threadClick della classe NewThread
    HelpFrame frameAiuto = new HelpFrame();
    Thread threadClick;

    // Avviare un file audio per segnalare il click del pulsante help
    // Inizializzazione di threadClick
    threadClick = new NewThread();
    // Resettaggio del nome di threadClick
    threadClick.setName("click");
    // Avvio di threadClick
    threadClick.start();

    // Settare il colore di sfondo del pannello principale contenuto in frameAiuto
    frameAiuto.getContentPane().setBackground(new Color(0, 102, 255));

    // Rendere visibile frameAiuto
    frameAiuto.setVisible(true);
} //GEN-LAST:event_helpActionPerformed
```

Inoltre, sempre nello StartFrame, sono stati settati i valori iniziali delle etichette, del punteggio, delle vite e del record massimo e anche settare inizialmente tutti i tool riguardanti l'inserimento del nome del giocatore invisibili:

```
// Settaggio iniziale dei valori delle etichette del punteggio, delle vite e del record massimo
framePrincipale.getPunteggio().setText("Punti: " + framePrincipale.getNPunti());
framePrincipale.getVite().setText("Vite: " + framePrincipale.getNVite());
framePrincipale.getRecordMassimo().setText("Record: " + framePrincipale.getPuntiRecord());

// Settare inizialmente come invisibili tutti i tool riguardanti l'inserimento del nome del giocatore
framePrincipale.getEtichettaInserimento().setVisible(false);
framePrincipale.getNomeGiocatore().setVisible(false);
framePrincipale.getConfirm().setVisible(true);
```

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Infine nella classe è stato scritto un blocco di codice che quando un giocatore vuole resettare la classifica, e quindi eliminare i dati al suo contenuto, non deve fare altro che cliccare un bottone per il reset e non appena viene cliccato si apre un popup in cui ti viene chiesta la conferma del reset. La funzione che svolge questa operazione è la `sb.append("AAA");` che scrive la stringa contenuta nelle virgolette al posto del vecchio nome del giocatore e del vecchio punteggio.

```
private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {GEN-FIRST:event_jButton1MouseClicked
    String nomeFile = "Classifica.csv", nomeGiocatore, punteggio, riga;
    String[] statoGiocatore;
    BufferedReader lettore;
    int i;
    int confermaReset = JOptionPane.showConfirmDialog(null, "Vuoi confermare il reset della classifica?", "Conferma", JOptionPane.YES_NO_OPTION);
    if(confermaReset == JOptionPane.YES_OPTION) {
        try{
            PrintWriter writer = new PrintWriter(new File(nomeFile));

            StringBuilder sb = new StringBuilder(); // ti permette di costruire una stringa
            sb.append("Giocatore");
            sb.append(';');
            sb.append("Punteggio");
            sb.append('\n');
            for(i=0; i<17; i++){
                sb.append("AAA");
                sb.append(';');
                sb.append("0");
                if(i < 17)
                    sb.append('\n');
            }

            writer.write(sb.toString());
            writer.close();

        }
        catch (FileNotFoundException e){
            JOptionPane.showMessageDialog(null, "File \"Classifica.csv\" assente.", "Errore", JOptionPane.ERROR_MESSAGE);
        }
        areaGiocatori1.setText("");
        areaPunteggi.setText("");
    }
}
```


Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

```
try {  
  
    // Inizializzare l'oggetto Lettore  
    lettore = new BufferedReader(new FileReader(nomeFile));  
  
    // Saltare la prima riga della tabella in "Classifica.csv" per evitare la lettura dei valori indesiderati presenti in questa  
    lettore.readLine();  
  
    // Leggere tante righe della tabella in "Classifica.csv" quante ce ne sono  
    while((riga = lettore.readLine()) != null) {  
  
        // Dividere la riga nei campi corrispondenti separandoli rispettivamente con un punto e virgola  
        statoGiocatore = riga.split(";");  
  
        // Assegnare l'elemento contenuto nel primo campo a nomeGiocatore  
        nomeGiocatore = statoGiocatore[0];  
  
        // Assegnare l'elemento contenuto nel secondo campo a punteggio  
        punteggio = statoGiocatore[1];  
  
        // Inserire i valori di nomeGiocatore e punteggio nelle apposite aree della classifica  
        areaGiocatori.append(nomeGiocatore + '\n');  
        areaPunteggi.append(punteggio + '\n');  
    }  
    // Gestire l'assenza del file richiamato  
} catch (FileNotFoundException ex) {  
    JOptionPane.showMessageDialog(null, "File \"Classifica.csv\" assente.", "Errore", JOptionPane.ERROR_MESSAGE);  
  
    // Gestire l'errore di lettura/scrittura del file richiamato  
} catch (IOException ex) {  
    JOptionPane.showMessageDialog(null, "Errore durante la lettura/scrittura del/sul file \"Classifica.csv\".", "Errore", JOptionPane.ERROR_MESSAGE);  
}
```

Ecco la guida di Jumping Ball che spiega come funziona il videogioco;

GUIDA JUMPING BALL

Non appena si avvia il gioco, quest'ultimo si presenta in tale modo, con una finestra rappresentante una classifica in posizione centrale che contiene i migliori 17 giocatori. Sotto questa classifica giace il tasto *Start* che, una volta cliccato, permette di avviare il gioco.



Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

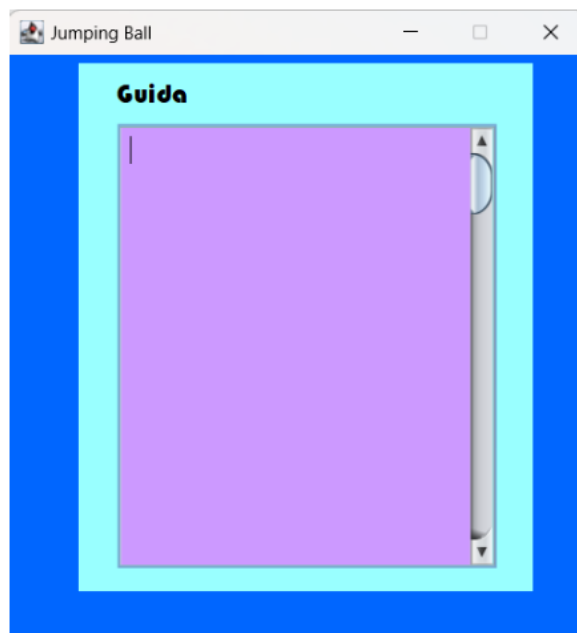
Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Tramite la pressione del pulsante che raffigura il cestino in basso a destra, è possibile resettare la classifica generale del gioco dopo aver cliccato su "Yes".



Invece premendo il pulsante raffigurante il punto interrogativo, sempre in basso a destra ma stavolta a destra del cestino, è possibile visualizzare la finestra della guida che spiega le istruzioni d'uso del videogioco.



Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

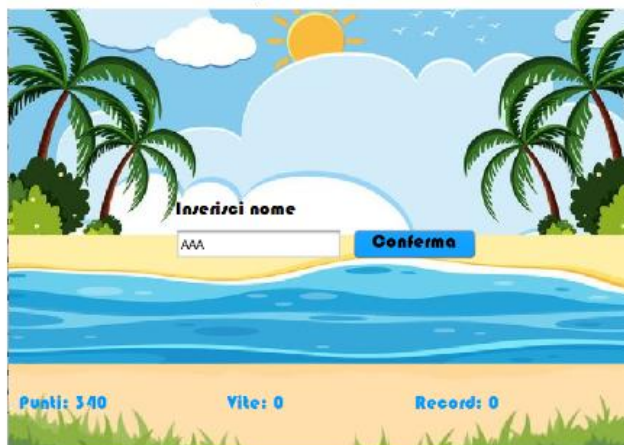
Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Una volta premuto il pulsante *Start* nella finestra iniziale, il gioco inizia dopo un countdown della durata di circa tre secondi. Subito dopo il conto alla rovescia, viene visualizzata la schermata di gioco.



Nel caso il giocatore fosse così scarso da perdere in tempi relativamente brevi, sarebbe costretto a inserire un nickname per registrare il suo punteggio in classifica. Dopo tale inserimento e una volta cliccato su *Conferma*, si ritorna nella schermata iniziale.



Classe: 4B ITT.

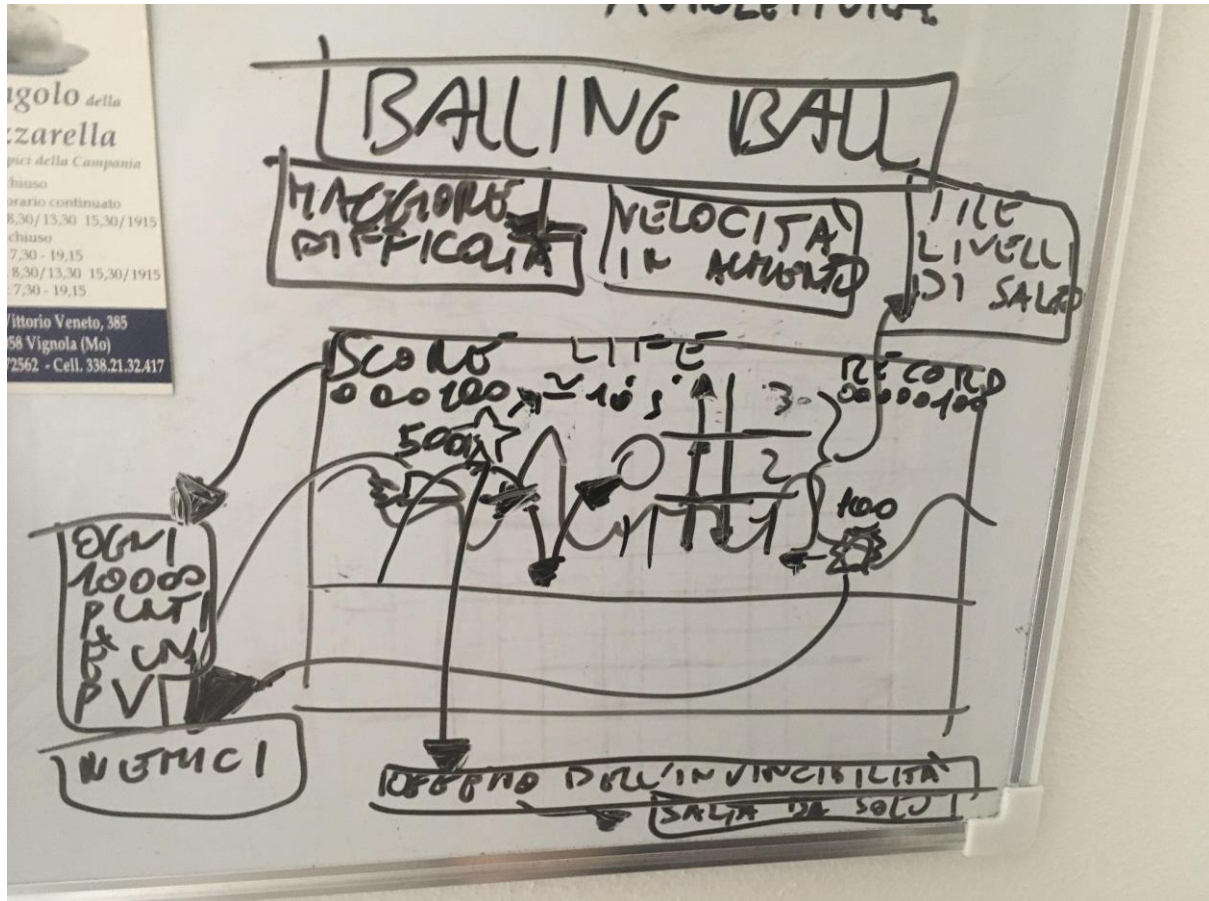
a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Ecco il work flow che era stato creato da Giuseppe come idea di partenza prima di creare il videogioco:



Classe: 4B ITT.

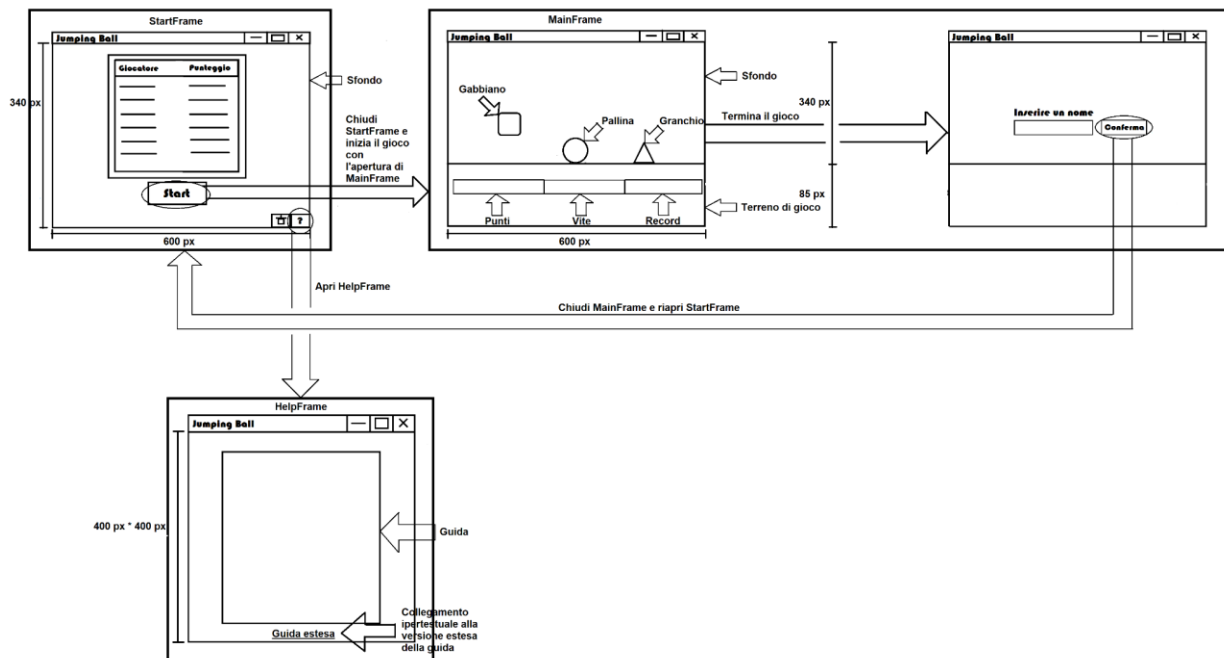
a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Qui invece si può notare l'immagine del progetto implementato in Paint che è stato fatto prima dell'inizio della creazione del gioco per avere un'idea generale di come dovesse essere il gioco una volta finito:



Infine elenchiamo tutti i vari link con i quali abbiamo trovato le giuste immagini e gif da implementare in Jumping Ball:

Immagine sfondo principale:

https://stock.adobe.com/search?k=ocean+clipart&asset_id=273578621

Immagine terreno di gioco:

<https://depositphotos.com/it/video/animation-of-tropical-landscape-beach-sea-waves-palms-air-plane-and-green-screen-51036185.html>

Immagine sfondo di inizio:

<https://it.vecteezy.com/arte-vettoriale/657284-scenario-di-cartone-animato-bellissima-spiaggia>

Immagine pallina:

https://www.flaticon.com/free-icon/tennis-ball_8686749

Classe: 4B ITT.

a.s.: 2023/24.

Titolo Progetto: Jumping Ball.

Nome del gruppo: Galaxy Java Development Team.

Componenti: Carlino Giuseppe; Grandi Elia; Sousane Souhaib.

Immagine granchio:

<https://it.cleanpng.com/png-8iluo6/download-png.html>

Immagine cestino: <https://pixabay.com/it/>

Immagine gabbiano: <https://corsopolaris.net/>