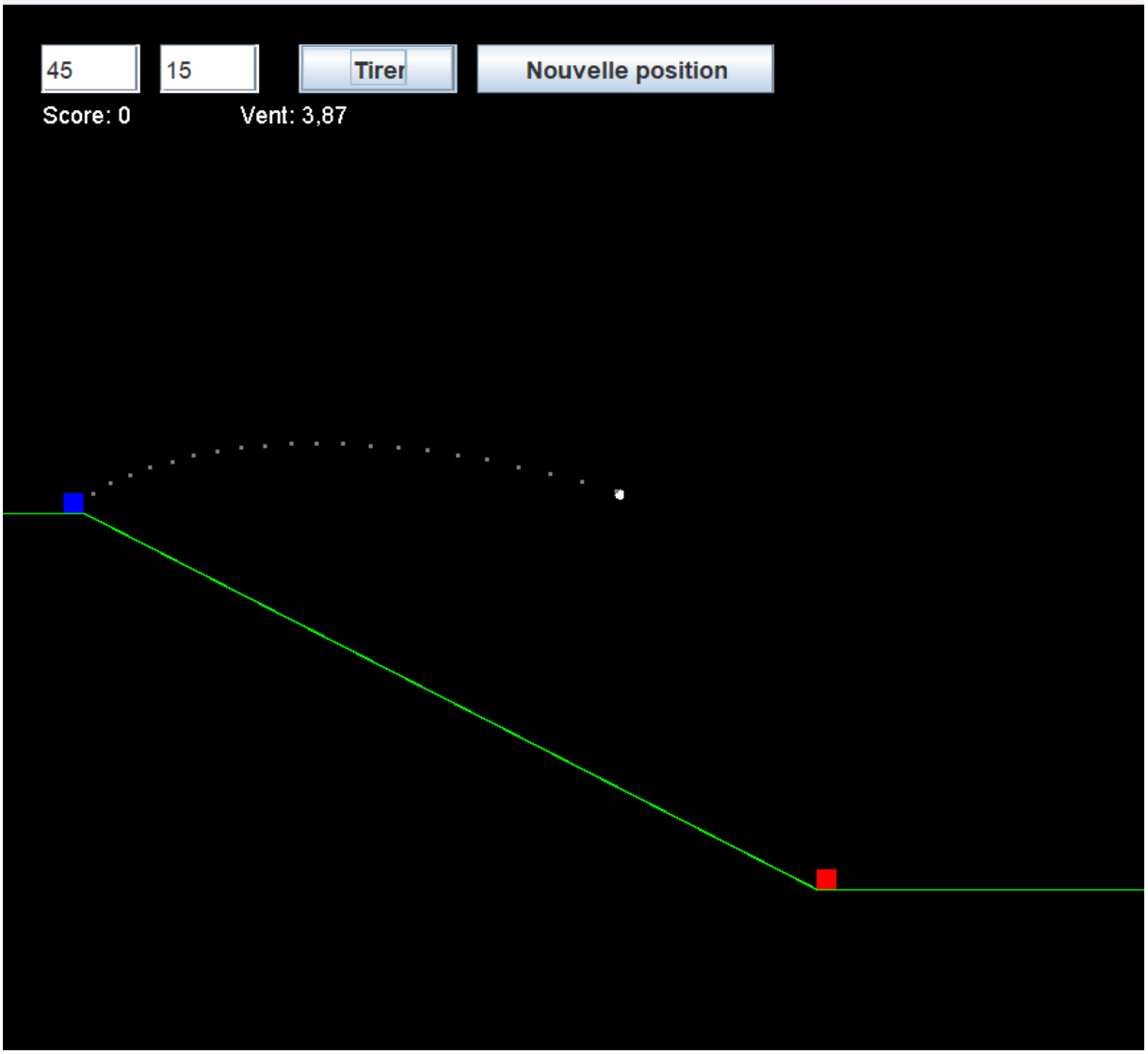
TP2 – TARGET

🎯 **Objectif général :**

Développer une application de simulation balistique :

- Gestion des données saisies au clavier
- Gestion des collisions et de la trace du projectile
- Gestion du score
- Mise en pratique du paquet Swing



Le canon est matérialisé par un carré bleu positionné aléatoirement dans la partie gauche de l'écran.
La cible est matérialisée par un carré rouge positionnée aléatoirement dans la partie droite de l'écran.

Le joueur saisit un angle de tir compris entre 0 et 90

Le joueur saisit une vitesse initiale comprise entre 10 et 100

Un bouton Nouvelle position permet de relancer le jeu.

Un bouton Tirer lance le tir balistique.

Le projectile, dont la trace est matérialisée à l'écran par des points est soumis

- A un vent horizontal aléatoire entre -5 (vers la gauche) et +5 (vers la droite)
- A la gravité selon l'axe vertical

Pour des raisons de simplification du modèle, on calculera uniquement la modification de la vitesse du projectile à chaque itération, ce qui donne sa future position :

```
// Vitesse initiale
vx = Math.cos(radians) * vitesse / 2;
vy = -Math.sin(radians) * vitesse / 2;
// où vitesse : la vitesse saisie par le joueur



// Itérations
// Nouvelles coordonnées du projectile :
projX += vx;
  vx += wind/10;
projY += vy;
  vy += gravity;

// La position du projectile est maintenant projX, projY
```

Conditions de sortie : le projectile sort de l'écran
Conditions de victoire : la cible est touchée par le projectile

📋 Étapes

1.1 Créer un fichier App.java

- Contient le point d'entrée de l'application
    - La méthode main instancie la classe jeu : GamePanel

```java
import javax.swing.*;

public class App {
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            JFrame frame = new JFrame("T.A.R.G.E.T.");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.setSize(600, 600);
            frame.setResizable(false);

            GamePanel gamePanel = new GamePanel();
            frame.add(gamePanel);
            frame.setVisible(true);
        });
    }
}
```

1.2 Créer un fichier GamePanel.java

```java
import java.awt.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import javax.swing.*;

public class GamePanel extends JPanel {
    private final Timer timer;
    private final Random random = new Random();
    //TODO
}
```

1.3 Créer les attributs de la classe GamePanel

- L'angle de tir (saisi par le joueur)
- La vitesse initiale (saisie par le joueur)
- La vitesse du vent (aléatoire entre – 5 et +5)
- Les coordonnées du canon (aléatoire)
- Les coordonnées de la cible (aléatoire)
- Les coordonnées du projectile (initialisées, puis calculées)

- Le score
- La trajectoire sous forme d'une liste de points ArrayList

1.4 Créer les méthodes de la classe GamePanel

Le constructeur

Définit les champs de saisie et les boutons
Calcule les positions du canon et de la cible
Calcule les vitesses initiales du projectile
Lance le timer (délai de 50 ms entre chaque itération

placeCannonAndTarget()

Calcule les coordonnées de la cible et du canon aléatoirement

startShooting()

Récupère et vérifie les valeurs saisies par l'utilisateur
Calcule les vitesses initiales du projectile

updateProjectile

Calcule les nouvelles coordonnées du projectile
Stocke la position dans la liste de points
Effectue un test de collision avec la cible
Effectue un test de sortie d'écran
Appelle de la méthode repaint() pour rafraichier l'écran.

Redéfinition de la méthode paintComponent()

```
@Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
    }
```

Affiche à l'écran le canon, la cible, le projectile, la trace du projectile, le score