TP1 – SNAKE

🎯 Objectif général :

Développer une application de jeu Snake :

- Gestion des événements clavier
- Gestion des collisions
- Gestion du score
- Mettre en pratique du paquet Swing

🧩 Partie 1 : Base de données et classe Utilisateur

📋 Étapes

1.1 Créer un fichier Snake.java

- Contient une seule classe Snake qui elle-même contient la méthode Main
    - La méthode main instancie la classe jeu : GameFrame

1.2 Créer un fichier GamaFrame.java

```
import javax.swing.JFrame;

public class GameFrame extends JFrame{
    GameFrame(){
        this.add(new GamePanel());
        this.setTitle("SNAKE");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setResizable(false);
        this.pack();
        this.setVisible(true);
        this.setLocationRelativeTo(null);
    }
}
```

1.3 Créer un fichier GamePanel.java

```
import java.awt.*;
import java.awt.event.*;
```

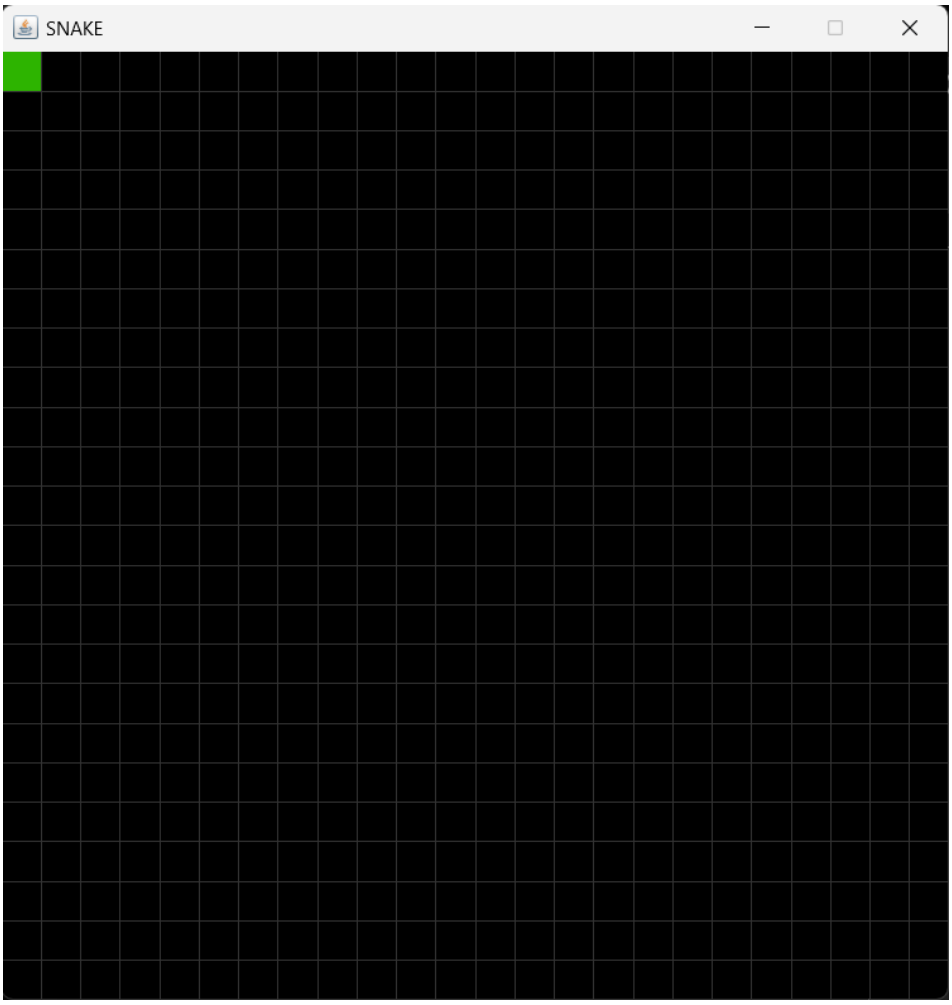
```java
import javax.swing.*;
import javax.swing.JPanel;
import java.util.Random;

public class GamePanel extends JPanel implements ActionListener {

}
```

1.4 Créer les attributs de la classe GamePanel

- Largeur d'écran et hauteur d'écran : 600 x 600
- L'écran est divisé en pas de 25 :



- Les coordonnées x et y de la tête du serpent sont des tableaux d'entiers de taille 600 / 25
- La taille initiale du serpent est de 6 cases (de 25 pixels)
- Une pomme est placée aux coordonnées appleX et appleY
- Une variable direction prenant les valeurs 'R', 'L' ,'U', 'D' selon la direction du serpent.

- La classe GamePanel inclue les méthodes :

StartGame()	Initialise le jeu en positionnant une pomme au hasard sur le plateau et démarre un timer
paintComponent(Graphics g)	Dessine à l'écran une pomme rouge (cercle). Dessine le serpent (n carrés verts). Affiche le score
newApple()	Calcule au hasard les coordonnées de la pomme appleX et appleY
move()	Gère le déplacement du serpent x et y en fonction de sa direction
checkApple()	Gère le contact de la tête du serpent avec la pomme. Si les coordonnées sont les mêmes : la taille du serpent augmente de 1, le score augmente de 1, une nouvelle pomme est placée sur le terrain
checkCollision()	Vérifie si la tête du serpent x[0], y[0] entre en collision avec le corps du serpent x[i], y[i] oi varie de 1 à taille du serpent. Vérifie aussi si la tête du serpent sort du terrain. Dans ces cas, le flag gameover est déclenché, le timer est arrêté.
actionPerformed(ActionEvent e)	Invoquée quand un événement est déclenché (ici par le timer). Appelle successivement les méthodes move(), checkApple(), checkCollisions(). Elle appelle enfin la méthode repaint() de swing pour rafraichir l'écran

.

et enfin la sous-classe :

```
    public class MyKeyAdapter extends KeyAdapter{
        @Override
        public void keyPressed(KeyEvent e){
            switch(e.getKeyCode()) {
                case KeyEvent.VK_LEFT:
                    if (direction != 'R') {
                        direction = 'L';
                    }
                    break;
                case KeyEvent.VK_RIGHT:
                    if (direction != 'L') {
                        direction = 'R';
                    }
                    break;
                case KeyEvent.VK_UP:
                    if (direction != 'D') {
                        direction = 'U';
```

```
                }
                break;
            case KeyEvent.VK_DOWN:
                if (direction != 'U') {
                    direction = 'D';
                }
                break;
        }
    }
```

Gère la saisie au clavier des touches de direction.