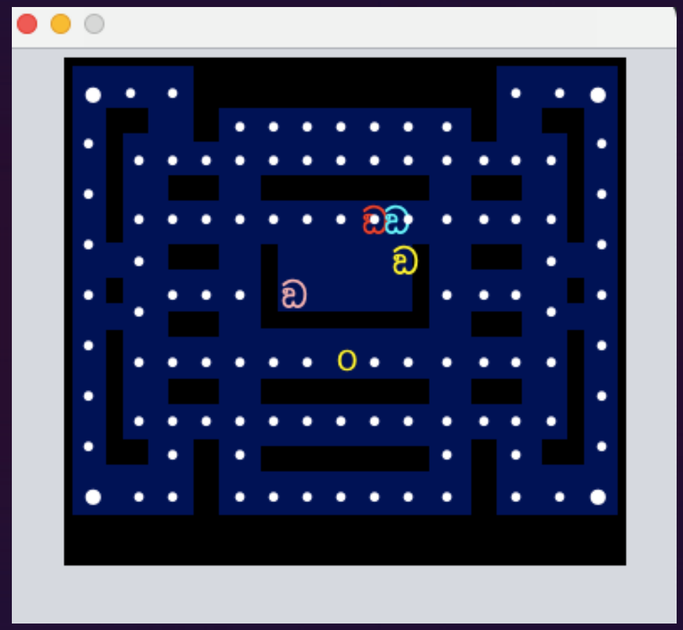

GAME
PROJECT
SIGN IN

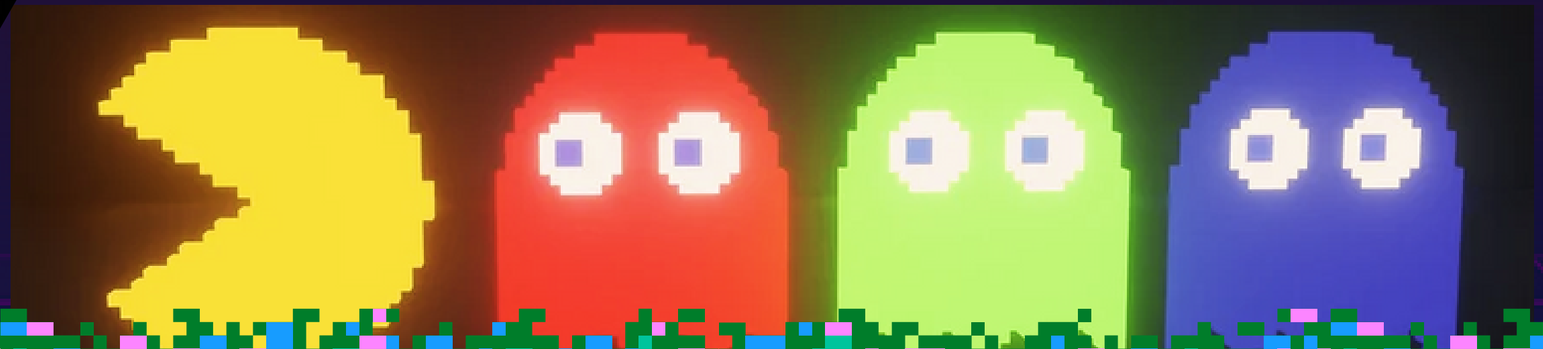
INTEGRANTES DEL EQUIPO

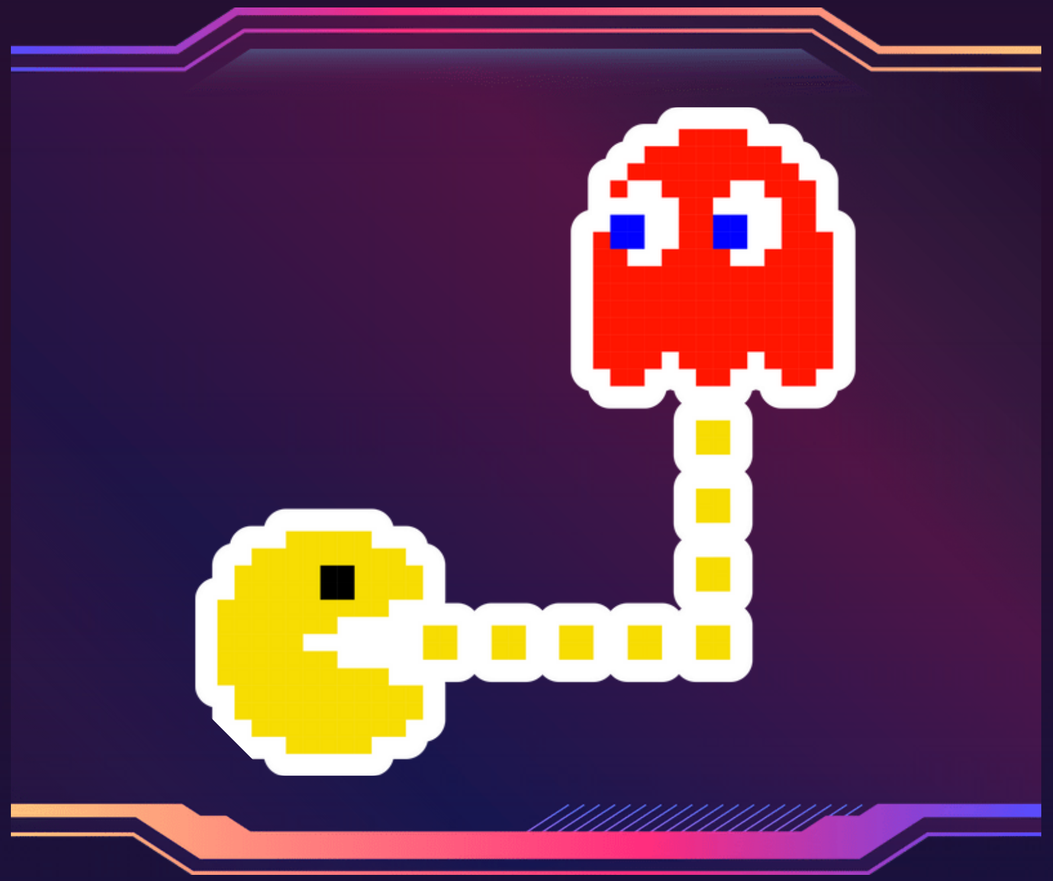
DIEGO OROZCO ALVARADO

SANTIAGO ELÍ JIMENEZ AGUÍLAR

PAC-MAN







CONOCIMIENTOS APLICADOS

Para la elaboración de este projecto tipo juego Pac-Man, hicimos uso de varios temas vistos en la materia, clases y objetos, arreglos, modificadores, herencias, etc. Cada uno ayudando a darle un buen camino al projecto.

CONOCIMIENTOS INVESTIGADOS

Lo principal que se tuvo que investigar fue el cómo lograr que el juego se pueda ver en pantalla, al igual que todos sus elementos, y el lograr procesos en paralelos.

CÓDIGO

SIMBOLOGÍA	
	EXTENDS
	IMPLEMENTS

KEYLISTENER

JFRAME

CHARACTER
ENVIRONMENT

RUNNABLE

CHARACTER
VENTANA

INICIAR

ACTUALIZADOR

ENEMYPOSITION

POINTSCONTROLLER

RANDOMENEMYMOVEMENT

ALGO DEL CODiGO

MOVEMENT

Este método recibe un objeto tipo Character. Y con el objeto y un numero random decidirá la dirección a la que se movera.

```
public static void movement(Character character) {
    Random random = new Random();
    switch (random.nextInt(bound:4)) {
        case 0:
            character.moveUp(character.x);
            break;
        case 1:
            character.moveDown(character.x, HEIGHT);
            break;
        case 2:
            character.moveRight(character.y, WIDTH);
            break;
        case 3:
            character.moveLeft(character.y);
            break;
        default:
            break;
    }
}
```

ALWAYSMOViNG

Este método guarda la posición actual del objeto Character y si en 50 milisegundos es la misma retorna true

```
public static boolean AlwaysMoving(Character character) {
    int FirstX, FirstY;
    FirstX=character.x;
    FirstY=character.y;
    try {
            Thread.sleep(millis:50);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    if (character.x==FirstX&&character.y==FirstY) {
        return true;
    }
    return false;
}
```

CREACIÓN DE LOS ELEMENTOS

Para la creación de los elementos que se ven en pantalla como Pac-Man, los fantasmas, los muros y puntos, se utilizaron labels con ”javax.swing.JLabel”

```
        //          Personajes:

//Jugador:
public javax.swing.JLabel jLabelPacMan= new javax.swing.JLabel();
protected static Character pacMan;

public javax.swing.JLabel jLabelPuntaje = new javax.swing.JLabel(); // puntaje
public javax.swing.JLabel jLabelVidas = new javax.swing.JLabel(); // vidas

//Fantasmas:
public javax.swing.JLabel jLabelGhostC= new javax.swing.JLabel(); // Cyan
public javax.swing.JLabel jLabelGhostP= new javax.swing.JLabel(); // pink
public javax.swing.JLabel jLabelGhostR= new javax.swing.JLabel(); // Red
public javax.swing.JLabel jLabelGhostY= new javax.swing.JLabel(); // Yellow
```

```
//              Muros:
public javax.swing.JLabel[] arrayJLabelMuros = new javax.swing.JLabel[35];
//              Puntos:
public javax.swing.JLabel[] arrayJLabelPuntos = new javax.swing.JLabel[115];
```

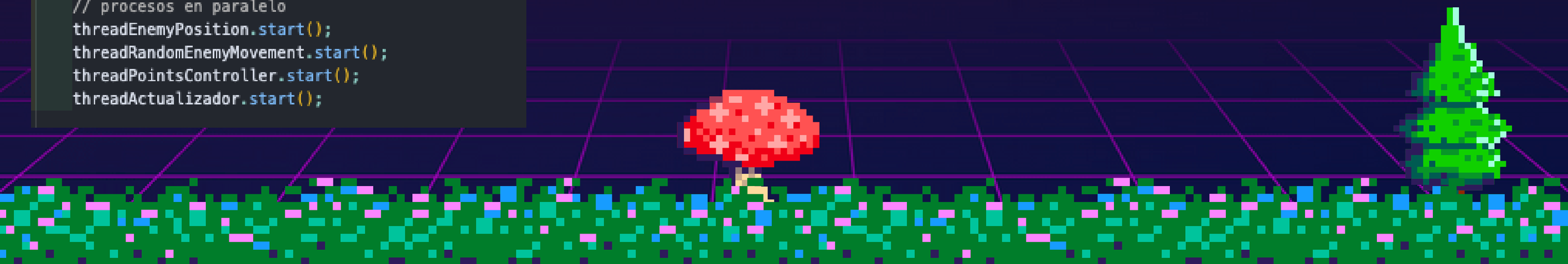
```
jLabelPacMan.setText(text:"O");
jLabelPacMan.setForeground(Color.YELLOW);    // PAC-MAN
jLabelPacMan.setFont(fuente2);


jLabelGhostR.setText(text:"ᗣ");
jLabelGhostR.setForeground(Color.RED);    // Ghost Red
jLabelGhostR.setFont(fuente);

jLabelGhostC.setText(text:"ᗣ");
jLabelGhostC.setForeground(Color.CYAN);    // Ghost Cyan
jLabelGhostC.setFont(fuente);

jLabelGhostP.setText(text:"ᗣ");
jLabelGhostP.setForeground(Color.PINK);    // Ghost Pink
jLabelGhostP.setFont(fuente);

jLabelGhostY.setText(text:"ᗣ");
jLabelGhostY.setForeground(Color.YELLOW);    // Ghost Yellow
jLabelGhostY.setFont(fuente);
```

USO DE HILOS

Utilizamos hilos para hacer procesos en paralelo como contar/agregar puntos mover los fantasmas y Pac-man indicar los movimientos de los fantasmas

```java
// hilos
Actualizador actualizador = new Actualizador();
Thread threadActualizador = new Thread(actualizador);// Crea actualizador (file) como un ejecutable
EnemyPosition enemyPosition = new EnemyPosition(ventana.jPanel1.getWidth(), ventana.jPanel1.getHeight());
Thread threadEnemyPosition = new Thread(enemyPosition); // Crea enemyPosition (file) como un ejecutable
RandomEnemyMovement randomEnemyMovement = new RandomEnemyMovement();
Thread threadRandomEnemyMovement = new Thread(randomEnemyMovement); // Crea randomEnemyMovement (file) como un ejecutable
PointsController pointsController = new PointsController();
Thread threadPointsController = new Thread(pointsController); // Crea pointsController (file) como un ejecutable
```

```java
// procesos en paralelo
threadEnemyPosition.start();
threadRandomEnemyMovement.start();
threadPointsController.start();
threadActualizador.start();
```



USO DE HILOS

```
public void run() {
    movement(ghostC);
    movement(ghostR);
    movement(ghostP);
    movement(ghostY);
    while (true) {
        if (AlwaysMoving(ghostC)==true) movement(ghostC);
        if (AlwaysMoving(ghostR)==true) movement(ghostR);
        if (AlwaysMoving(ghostP)==true) movement(ghostP);
        if (AlwaysMoving(ghostY)==true) movement(ghostY);
    }
}
}
```

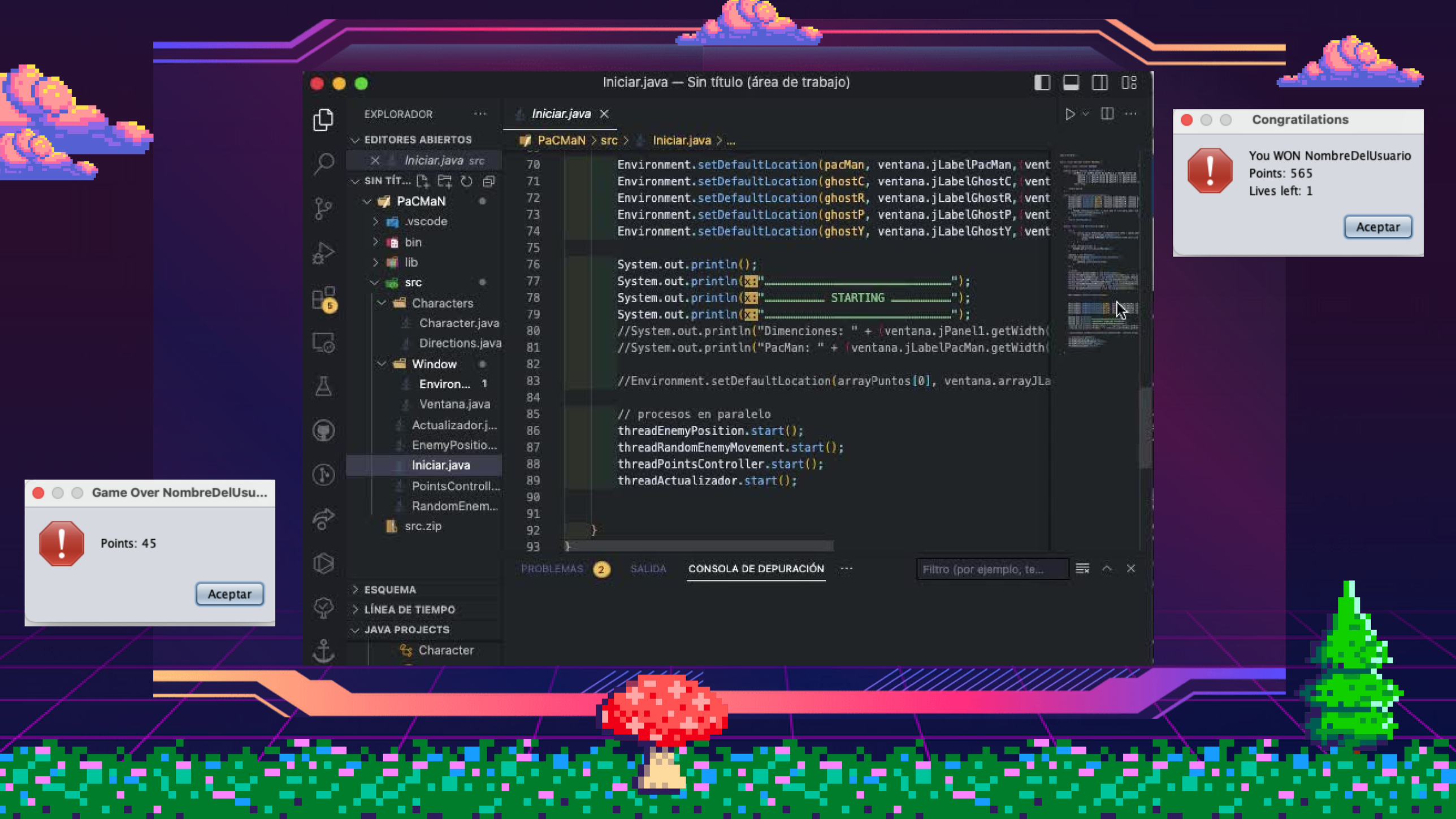
```
    @Override
public void run() {
    while (true) {
        EnemyPosition.movement(ghostC);
        EnemyPosition.movement(ghostR);
        EnemyPosition.movement(ghostP);
        EnemyPosition.movement(ghostY);
        try {
            Thread.sleep(millis:3000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

```
    @Override
    public void run() {
        while(true){
            pointsChecker(Iniciar.ventana,arrayJLabelPuntos);

        }
    }

}
```

```
public void run() {
    System.out.println("vidas = " + vidas);
    while (vidas>=0) {
        Iniciar.actualizarLabel();


        if (pacMan.Colicion(ghostC.x, ghostC.y, jLabelGhostC.getWidth(), (jLabelGhostC.getHeight()-8)) == true){
            if(ghostC.getKillable()==true){
                Environment.setDefaultLocation(ghostC, jLabelGhostC,(jPanel1.getWidth()/2-5),y:125);
                ghostC.setKillable(flag:false);
                Iniciar.ventana.jLabelGhostC.setForeground(Color.CYAN);
            }else{
                vidas--;
                Environment.setDefaultLocation(pacMan, jLabelPacMan,(jPanel1.getWidth()/2-5),y:178);
                if (vidas>=0)System.out.println("Game Over Ghost C : vidas = " + vidas);
            }
        }
        if (pacMan.Colicion(ghostR.x, ghostR.y, jLabelGhostR.getWidth(), jLabelGhostR.getHeight()-8) == true){
            if(ghostR.getKillable()==true){
                Environment.setDefaultLocation(ghostR, jLabelGhostR,(jPanel1.getWidth()/2-5),y:125);
                ghostR.setKillable(flag:false);
                Iniciar.ventana.jLabelGhostR.setForeground(Color.RED);
            }else{
            vidas--;
            Environment.setDefaultLocation(pacMan, jLabelPacMan,(jPanel1.getWidth()/2-5),y:178);
            if (vidas>=0)System.out.println("Game Over Ghost R : vidas = " + vidas);
            }
        }
        if (pacMan.Colicion(ghostP.x, ghostP.y, jLabelGhostP.getWidth(), jLabelGhostP.getHeight()-8) == true){
            if(ghostP.getKillable()==true){
                Environment.setDefaultLocation(ghostP, jLabelGhostP,(jPanel1.getWidth()/2-5),y:125);
                ghostP.setKillable(flag:false);
                Iniciar.ventana.jLabelGhostP.setForeground(Color.PINK);
            }else{
            vidas--;
            Environment.setDefaultLocation(pacMan, jLabelPacMan,(jPanel1.getWidth()/2-5),y:178);
            if (vidas>=0)System.out.println("Game Over Ghost P : vidas = " + vidas);
            }
        }
        if (pacMan.Colicion(ghostY.x, ghostY.y, jLabelGhostY.getWidth(), jLabelGhostY.getHeight()-8) == true){
            if(ghostY.getKillable()==true){
                Environment.setDefaultLocation(ghostY, jLabelGhostY,(jPanel1.getWidth()/2-5),y:125);
                ghostY.setKillable(flag:false);
                Iniciar.ventana.jLabelGhostY.setForeground(Color.YELLOW);
            }else{
            vidas--;
            Environment.setDefaultLocation(pacMan, jLabelPacMan,(jPanel1.getWidth()/2-5),y:178);
            if (vidas>=0)System.out.println("Game Over Ghost Y : vidas = " + vidas);
            }
        }
    }
    JOptionPane.showMessageDialog(jPanel1, "Points: "+points, title:"Game Over", messageType:0);
    System.exit(status:0);// cierra el programa
    }
}
```

START THE GAME

START


Iniciar.java — Sin título (área de trabajo)
EXPLORADOR
EDITORES ABIERTOS
Iniciar.java src
SIN TÍT...
PaCMaN
.vscode
bin
lib
src
Characters
Character.java
Directions.java
Window
Environ... 1
Ventana.java
Actualizador.j...
EnemyPositio...
Iniciar.java
PointsControll...
RandomEnem...
src.zip
ESQUEMA
LÍNEA DE TIEMPO
JAVA PROJECTS
Character
Iniciar.java
PaCMaN > src > Iniciar.java > ...
70 Environment.setDefaultLocation(pacMan, ventana.jLabelPacMan, vent
71 Environment.setDefaultLocation(ghostC, ventana.jLabelGhostC, vent
72 Environment.setDefaultLocation(ghostR, ventana.jLabelGhostR, vent
73 Environment.setDefaultLocation(ghostP, ventana.jLabelGhostP, vent
74 Environment.setDefaultLocation(ghostY, ventana.jLabelGhostY, vent
75
76 System.out.println();
77 System.out.println("----------");
78 System.out.println("---------- STARTING ----------");
79 System.out.println("----------");
80 //System.out.println("Dimenciones: " + ventana.jPanel1.getWidth(
81 //System.out.println("PacMan: " + ventana.jLabelPacMan.getWidth(
82
83 //Environment.setDefaultLocation(arrayPuntos[0], ventana.arrayJLa
84
85 // procesos en paralelo
86 threadEnemyPosition.start();
87 threadRandomEnemyMovement.start();
88 threadPointsController.start();
89 threadActualizador.start();
90
91
92 }
93 }
PROBLEMAS 2
SALIDA
CONSOLA DE DEPURACIÓN
Filtro (por ejemplo, te...
Congratilations
You WON NombreDelUsuario
Points: 565
Lives left: 1
Aceptar
Game Over NombreDelUsu...
Points: 45
Aceptar