```java
package juego;

import java.awt.Color;

public class Arbol {

    private double x;
    private double y;
    private double ancho;
    private double alto;
    private double angulo;
    private Color color;
    private double escala;
    private Image imagen;
    private Image imagen2;
    private boolean dioPuntos;
    private boolean conSerpiente;

    public Arbol(double x, double y, double escala) {
        this.x = x;
        this.y = y;
        this.ancho = 130;
        this.alto = 5;
        this.angulo = 0;
        this.color = Color.MAGENTA;
        this.escala = escala;
        this.imagen = Herramientas.cargarImagen("arbol.png");
        this.imagen2 = Herramientas.cargarImagen("rama.png");
        this.dioPuntos=false;
        this.conSerpiente = false;
    }// Arbol




    public void dibujarArbol(Entorno e) {
        e.dibujarImagen(imagen, this.x, this.y, angulo, this.escala);
//      e.dibujarRectangulo(this.x, this.y, this.ancho, this.alto, this.angulo, this.color);

        e.dibujarImagen(imagen2, this.x, this.y - 14, angulo, .15);
    }// crearArbol

    public static boolean arbolesVacios(Arbol[] a) {
        int contador = 0;
        for (int i = 0; i < a.length; i++) {
            //si el arbol es null, contador aumenta
            if (a[i] == null) {
                contador++;
            }
        }
        // si contador es igual al numero de objetos arbol retorna true
        if (contador == a.length) {
            return true;
        } else {
            return false;
        }
    }// arbolesVacios

    public boolean isConSerpiente() {
        return conSerpiente;
    }



    public void setConSerpiente(boolean conSerpiente) {
        this.conSerpiente = conSerpiente;
    }



    public static Arbol[] crearArboles(Arbol[] arboles, Entorno e) {
        Random random = new Random();
        // si arbolesVacios es true se podra recorrer el for
        if (arbolesVacios(arboles)) {
            int n = 0;
            for (int i = 0; i < arboles.length; i++) {
```

```
                //si el arbol es igual a null se crea un random y se podra recorrer el while
                if (arboles[i] == null) {
                    // se crea un altura aleatoria para las ramas de los arboles
                    int rand = random.nextInt(300, 350);
                    while(rand%5==0) {
                        rand = random.nextInt(300, 350);
                    }
                    // se crea las distancia para los arboles fuera de la pantalla (lado derecho)
                    int rand2 = random.nextInt(10);
                    // escala para los arboles
                    double escala = Math.round(random.nextDouble(0.08, 0.12) * 100.0) / 100.0;
                    arboles[i] = new Arbol(e.ancho() + n + rand2, rand, escala);
                    n += 250;
                } // if
            } // for
            return arboles;
        } else {
            for (int i = 0; i < arboles.length; i++) {
                // si arbol en la posicion 0 es null se crea un random y se podra recorrer el while
                if (arboles[0] == null) {
                    int rand = random.nextInt(300, 350);
                    while(rand%5==0) {
                        rand = random.nextInt(300, 350);
                    }
                    int rand2 = random.nextInt(100);
                    double escala = Math.round(random.nextDouble(0.08, 0.12) * 100.0) / 100.0;
                    // como arboles[0] no tiene un elemento previo entonces se accede al ultimo
                    // elemento
                    double ultimoX = arboles[arboles.length - 1].x;
                    arboles[0] = new Arbol(ultimoX + e.ancho(), rand, escala);
                } else {
                    //si el arbol es null se crea un random y se podra recorrer el while
                    if (arboles[i] == null) {
                        int rand = random.nextInt(300,350);
                        while(rand%5==0) {
                            rand = random.nextInt(300, 350);
                        }
                        int rand2 = random.nextInt(200,400);
                        double escala = Math.round(random.nextDouble(0.08, 0.12) * 100.0) / 100.0;

                        // se accede al arbol anterior para obtener su x
                        arboles[i] = new Arbol(arboles[i - 1].x + rand2, rand, escala);
                    }
                }
            }
            return arboles;
        }

    }// crearArboles


    public boolean saleDePantalla() {
        if (this.x < -100) {
            return true;
        } else {
            return false;
        }
    }

    public void desplazar(double v) {
        this.x -= v;
    }// desplazar


    public double getAncho() {
        return ancho;
    }

    public double getAlto() {
        return alto;
    }

    public double getX() {
        return x;
    }

    public double getY() {
```

```
160         return y;
161     }
162 
163     public boolean isDioPuntos() {
164         return dioPuntos;
165     }
166 
167 
168 
169     public void setDioPuntos(boolean dioPuntos) {
170         this.dioPuntos = dioPuntos;
171     }
172 }// class Arbol
```