```java
package juego;

import java.awt.Color;

public class Escaleras {

    int pos;
    double x;
    double y;
    double ancho;
    double alto;

    public Escaleras(int pos, Viga[] suelos) {

        Random rnd = new Random();
        int offsetEscalera = rnd.nextInt(50);

        if (pos % 2 == 0) {
            this.x = suelos[pos + 1].extremoDerecho() - 30 - offsetEscalera;
        } else {
            this.x = suelos[pos + 1].extremoIzquierdo() + 30 + offsetEscalera;
        }

        this.y = ((suelos[pos].dondeEmpiezaElSuelo() - suelos[pos + 1].dondeEmpiezaElSuelo()) / 2)
                + suelos[pos + 1].dondeEmpiezaElSuelo();
        this.ancho = 30;
        this.alto = suelos[pos].dondeEmpiezaElSuelo() - suelos[pos + 1].dondeEmpiezaElSuelo();

    }

    public void dibujar(Entorno entorno) {

        // Rectángulo básico de la viga, respetando los valores indicados por el
        // constructor
        entorno.dibujarRectangulo(this.x, this.y, this.ancho, this.alto, 0.0, Color.BLUE);

        double paso = this.y + (this.alto / 2) - 3;

        // Se decide que la suma de la base de un triangulo, la punta del triángulo
        // adyacente y un espacio
        // extra sea la 25ava parte del ancho de la viga - 4 pixeles
        double rectangulos = (this.alto / 10);

        // Indica la cantidad de parejas de triangulos dibujados. Una pareja es un
        // triangulo con la punta hacia arriba
        // y el otro con la punta hacia abajo.
        int dibujados = 0;

        // Este bucle dibuja la pareja de triángulos a lo largo de la viga.

        while (dibujados <= rectangulos) {

            entorno.dibujarRectangulo(this.x, paso, 28, 9, 0.0, java.awt.Color.BLACK);
            paso -= 10;

            dibujados += 1;

        }

    }
```

```
65
66      public int lateralDerecho() {
67          return (int) this.x + 15;
68      }
69
70      public int lateralIzquierdo() {
71          return (int) this.x - 15;
72      }
73
74      public int extremoSuperior() {
75          return (int) (this.y - (this.alto / 2));
76      }
77
78      public int extremoInferior() {
79          return (int) (this.y + (this.alto / 2));
80      }
81
82  }
83
```