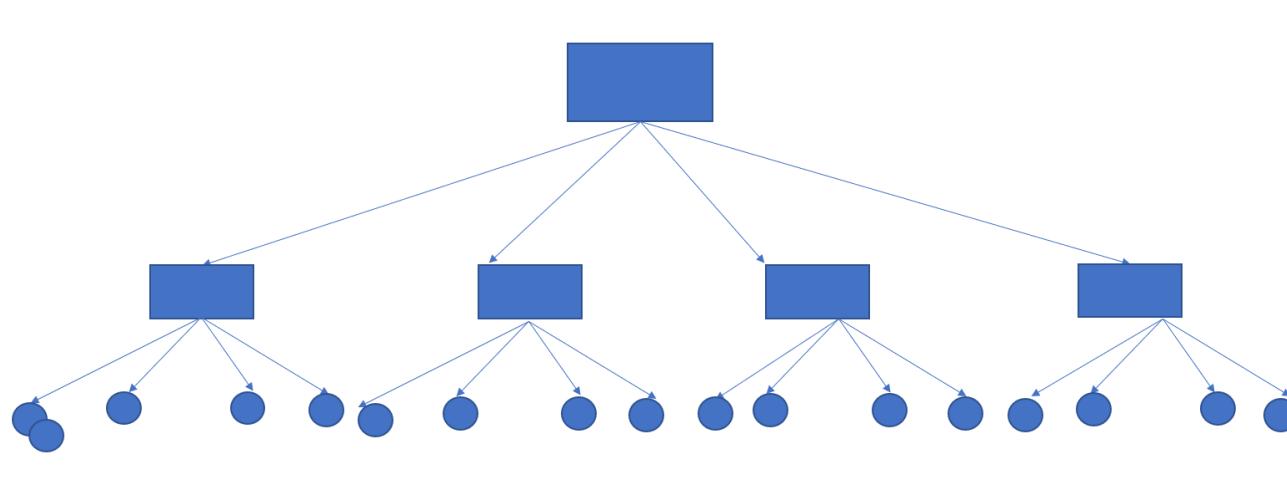


# **ALGORITMO PARA PREVENIR COLICIONES ENTRE ABEJAS ROBOTICAS**

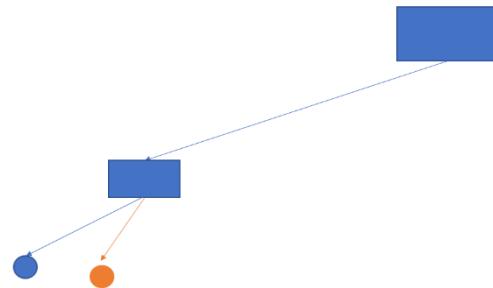
*Juan Camilo Guerrero Alarcon  
Santiago Pulgarin Vasquez  
Medellín, 6 de Noviembre de 2018*

# QuadTree



**Gráfico 1:** QuadTree de abejas. Una abeja es una clase que contiene coordenada “x” y coordenada “y”

# *Operaciones de la Estructura de Datos*



**Gráfico 2:** Operación de insertado  
de una estructura de datos

Clase	Complejidad
Abeja	$O(1)$
AgregarAbejas	$O(n)$
Boundary	$O(1)$
Main	$O(1)$
QuadTree	$O(n)$

**Tabla 1:** Complejidad de las operaciones  
de la estructura de datos

# *Criterios de Diseño de la Estructura de Datos*

- La razón por la cual escogimos la estructura de datos QuadTree es debido a que es una de las más usadas cuando se tratan problemas de ubicación espacial, más conocidas y muy acertada para tipos de problemas como este, además de su excelente implementación que facilita el análisis de los datos.

# *Consumo de Tiempo y Memoria*

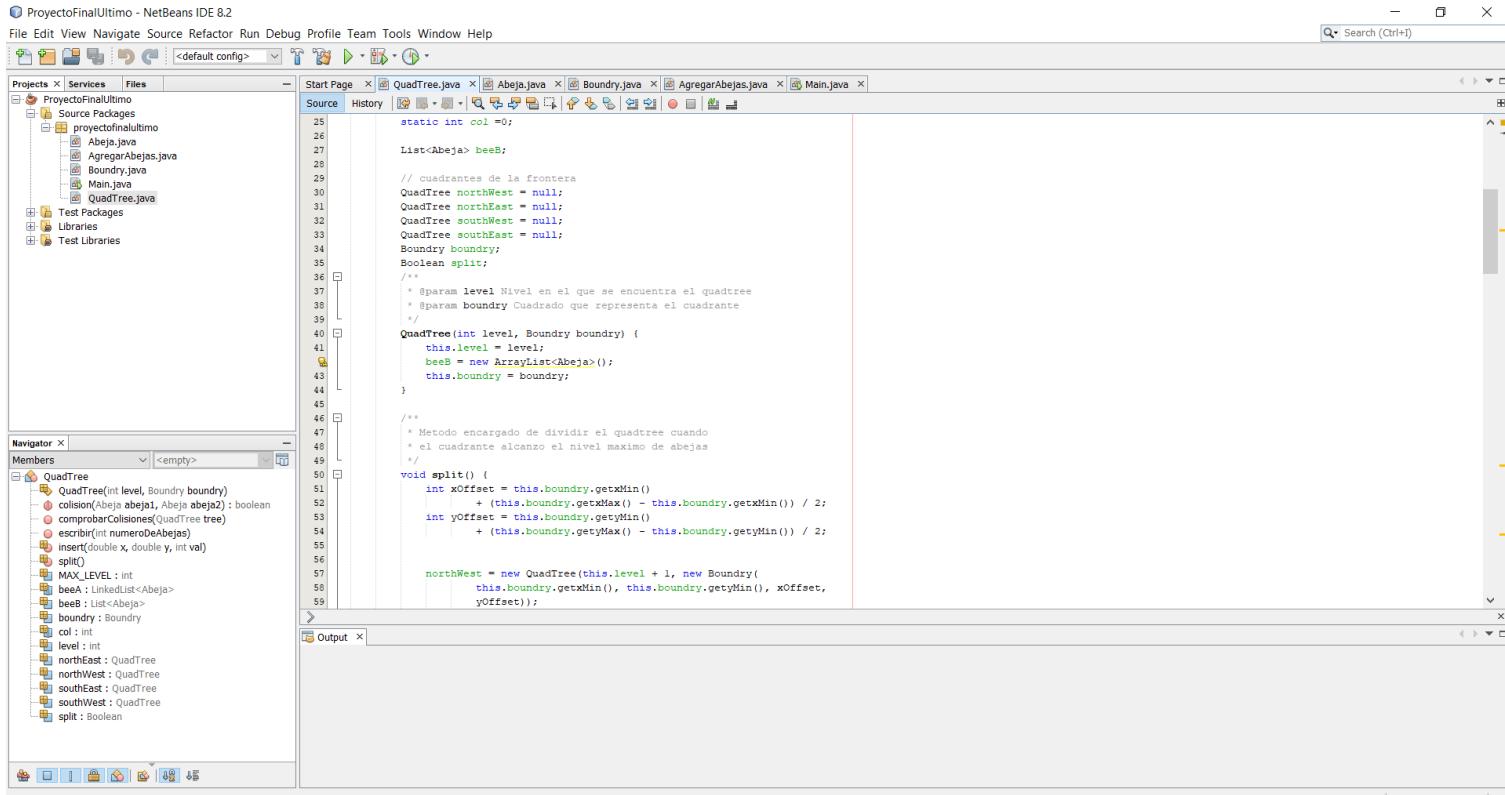
	CONJUNTO DE DATOS 1 (10)	CONJUNTO DE DATOS 2 (100)	CONJUNTO DE DATOS 3 (1000)	CONJUNTO DE DATOS 4 (100000)	CONJUNTO DE DATOS 5 (1000000)
TIEMPO QUE TARDO	0,000001 segundos	0,0008 segundos	0,009 segundos	0,10 segundos	10 segundos

**Tabla 2:** Tiempos de ejecución de las operaciones de la estructura de datos con diferentes conjuntos de datos

	Conjunto de datos1 (10)	CONJUNTO DE DATOS 2 (100)	CONJUNTO DE DATOS 3 (1000)	CONJUNTO DE DATOS 4 (100000)	CONJUNTO DE DATOS 5 (1000000)
Consumo de memoria	11	42	100	170	620

**Tabla 3:** Consumo de memoria de la estructura de datos con diferentes conjuntos de datos

# Software Desarrollado



The screenshot shows the NetBeans IDE 8.2 interface with the following details:

- Project:** ProyectoFinalUltimo
- Source Packages:** proyectofinalultimo (containing Abeja.java, AgregarAbejas.java, Boundary.java, Main.java, and QuadTree.java).
- Source Editor:** The main window displays the `QuadTree.java` file. The code implements a QuadTree data structure for managing bees. It includes methods for collision detection, inserting bees, and splitting quadrants. The code uses Java annotations like `@Override` and `@NotNull`.
- Navigator:** Shows the class hierarchy and member variables for `QuadTree`.
- Output:** An empty output pane.

```
static int col =0;
List<Abeja> beeB;
// cuadrantes de la frontera
QuadTree northWest = null;
QuadTree northEast = null;
QuadTree southWest = null;
QuadTree southEast = null;
Boundary boundary;
Boolean split;
/**
 * Guarda level Nivel en el que se encuentra el quadtree
 * Guarda boundary Cuadrado que representa el cuadrante
 */
QuadTree(int level, Boundary boundary) {
    this.level = level;
    beeB = new ArrayList<Abeja>();
    this.boundary = boundary;
}

/**
 * Metodo encargado de dividir el quadtree cuando
 * el cuadrante alcanzo el nivel maximo de abejas
 */
void split() {
    int xOffset = this.boundary.getMinX()
        + (this.boundary.getMaxX() - this.boundary.getMinX()) / 2;
    int yOffset = this.boundary.getMinY()
        + (this.boundary.getMaxY() - this.boundary.getMinY()) / 2;

    northWest = new QuadTree(this.level + 1, new Boundary(
        this.boundary.getMinX(), this.boundary.getMinY(), xOffset,
        yOffset));
}
```

Gráfico 4: Estructura de datos QuadTree