```java
package Tarefa;

import Auxiliar.Direccao;

public class DetectarRobot extends TarefaRobot {

    /**
     * variavel usada para contador
     */
    private int counter;

    /**
     * variavel com um conjunto de direccoes onde possivelmente está um obstaculo
     */
    private int[] potencial;

    /**
     * constantes com os valores definidos para obstaculos
     */
    private static final int limiarDeDeteccao = 4;
    private static final int toleranciaMudanca = 4;
    private static final int limiarDeObstaculo = 14;


    /**
     * Metodo Construtor
     * @param robot HemissonRobot corrente
     * @param aval AvaliadorAmbiente corrente
     */
    public DetectarRobot(HemissonRobot robot, AvaliadorAmbiente aval) {
        super(robot,aval);
        valoresSensores = robot.readSonar();
        this.potencial = null;
    }

    /**
     * Metodo para a deteccao de obstaculos
     * @param sensorIndex
     * @param sensorValue
     * @return boolean detect?
     */
    private boolean detect(int sensorIndex, int sensorValue) {
        this.counter = 0;
        readValues();
        return detect_recursive(sensorIndex, sensorValue);
    }


    /**
     * Metodo para a deteccao de obstaculos
     * @param sensorIndex
     * @param sensorValue
     * @return boolean
     */
    private boolean detect_recursive(int sensorIndex, int sensorValue) {
        robot.beeper(1);
        if(this.counter >= DetectarRobot.limiarDeDeteccao)
            return false;
        else if(Math.abs(this.valoresSensores[sensorIndex] - sensorValue) >
 DetectarRobot.toleranciaMudanca)
            return true;
        else {
            readValues();
            this.counter++;
            return detect_recursive(sensorIndex, sensorValue);
        }
```

```
    }

    private void readValues() {
        this.valoresSensores = robot.readSonar();
    }

    public void runOnce() throws Exception{
        robot.setSpeed(0,0);
        System.out.println(robot.getNome() + ": Estou parado");
        readValues();
        potencial = scan();

        if(potencial != null) {
            //Existe um obstaculo algures... talvez seja um "amigo"
            if(detect(potencial[0],potencial[1])) {
                System.out.println("Detectei um amiguinho na direccao: " +
Direccao.getText(potencial[0]));
                robot.beeper(5);
                System.exit(1);
            } else {
                System.out.println("Detetei um obstaculo na direccao: " +
Direccao.getText(potencial[0]));
                Direccao.getDireccaoOposta(new
Direccao(potencial[0])).executarMovimento(robot);
                robot.setSpeed(5,5);
                Thread.sleep(600);
                robot.setSpeed(0,0);
            }
        }

        passear();
        Thread.sleep(300);
    }

    private void passear() {
        robot.setSpeed(3,3);
    }

    private int[] scan() {
        if(valoresSensores == null)
            return null;

        int[] resposta = {-1,-1};
        for(int i = 0; i<valoresSensores.length; i++)
            if(i != ValoresSensores.chaoDireita && i != ValoresSensores.chaoEsquerda &&
valoresSensores[i] >= limiarDeObstaculo && valoresSensores[i] > resposta[1]) {
                resposta[0] = i;
                resposta[1] = valoresSensores[i];
            }
        if(resposta[0] == -1)
            return null;
        else
            return resposta;
    }
}
```