```java
package Tarefa;

import hemissonComm.HemissonRobot;

public class PassearFugirDetect extends TarefaRobot {
    private int counter;
    private int[] potencial;

    private static final int limiarDeDeteccao = 4;
    private static final int toleranciaMudanda = 4;
    private static final int limiarDeObstaculo = 14;

    public PassearFugirDetect(HemissonRobot robot, AvaliadorAmbiente aval) {
        super(robot,aval);
        this.counter = 0;
        this.potencial = null;
    }

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

    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;
    }

    public void runOnce() throws Exception {
        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 {
                Direccao paraOndeIr = Direccao.getDireccaoOposta(potencial[0]);
                paraOndeIr.executarMovimento(robot);
                robot.setSpeed(6,6);
            }
        }

        robot.setSpeed(3,3);
    }

    //Entry Point to recursive detection
    private boolean detect(int sensorIndex, int sensorValue) {
        robot.setSpeed(0,0);
        this.counter = 0;
        readValues();
        return detect_recursive(sensorIndex, valoresSensores[sensorIndex]);
```

```
    }

    //Recursive Method
    private boolean detect_recursive(int sensorIndex, int sensorValue) {
        robot.beeper(1);
        if(this.counter >= PassearFugirDetect.limiarDeDeteccao)
            return false;
        else if(Math.abs(this.valoresSensores[sensorIndex] - sensorValue) >
 PassearFugirDetect.toleranciaMudanda)
            return true;
        else {
            readValues();
            this.counter++;
            return detect_recursive(sensorIndex, sensorValue);
        }

    }

}
```