```java
package Tarefa;

import hemissonComm.HemissonRobot;
import java.util.ArrayList;
import Auxiliar.Direccao;
import Auxiliar.ValoresSensores;
import Avaliador.AvaliadorAmbiente;

public class PassearFugir extends TarefaRobot {

    private Direccao[] obstaculos;

    private static final int obstaculoLimiar = 20;

    public PassearFugir(HemissonRobot robot, AvaliadorAmbiente aval) {
        super(robot,aval);
        this.obstaculos = null;
    }

    public void runOnce() throws Exception{
        valoresSensores = robot.readSonar();
        obstaculos = avaliarSorrund(valoresSensores);
        fugirObstaculos(obstaculos,valoresSensores).executarMovimento(robot);
        robot.setSpeed(3,3);
    }

    private Direccao fugirObstaculos(Direccao[] obstaculos, int[] valoresSensores) {
        if(obstaculos.length == 0)
            return new Direccao(Direccao.frente);

        ArrayList <Direccao> todas = new ArrayList<Direccao>();
        todas.add(new Direccao(Direccao.frente));
        todas.add(new Direccao(Direccao.tras));
        todas.add(new Direccao(Direccao.esquerda));
        todas.add(new Direccao(Direccao.direita));
        todas.add(new Direccao(Direccao.fd));
        todas.add(new Direccao(Direccao.fe));

        int menor = findMenor();
        return Direccao.efectuaTraducao(menor);
    }

    private int findMenor() {
        int menor = valoresSensores[0];
        int mp = 0;
        for(int i = 1; i<valoresSensores.length; i++)
            if(valoresSensores[i] < menor)
            {
                menor = valoresSensores[i];
                mp = i;
            }

        return mp;
    }

    private Direccao[] avaliarSorrund(int[] valoresSensores) {
        ArrayList <Direccao> lista = new ArrayList<Direccao>();
        if(valoresSensores[ValoresSensores.frente] >= obstaculoLimiar)
            lista.add(new Direccao(Direccao.frente));
        if(valoresSensores[ValoresSensores.tras] >= obstaculoLimiar)
            lista.add(new Direccao(Direccao.tras));
        if(valoresSensores[ValoresSensores.frenteDireita] >= obstaculoLimiar)
            lista.add(new Direccao(Direccao.fd));
        if(valoresSensores[ValoresSensores.direita] >= obstaculoLimiar)
            lista.add(new Direccao(Direccao.direita));
        if(valoresSensores[ValoresSensores.frenteEsquerda] >= obstaculoLimiar)
```

```
            lista.add(new Direccao(Direccao.fe));
        if(valoresSensores[ValoresSensores.esquerda] >= obstaculoLimiar)
            lista.add(new Direccao(Direccao.esquerda));

        Direccao[] retorno = new Direccao[lista.size()];
        for(int i = 0; i<lista.size(); i++)
            retorno[i] = lista.get(i);

        return retorno;
    }

}
```