14:06

MeridianPrime.java

```java
import java.util.Random;
import java.util.concurrent.*;

public class MeridianPrime {

    private static final Random random = new Random();

    public static void main(String[] args) throws Exception {

        ExecutorService executor = Executors.newFixedThreadPool(5);

        System.out.println("🛰️ Iniciando monitoreo de sistemas en Meridian Prime...\n");

        Callable<Boolean> trafico = new Callable<Boolean>() {
            public Boolean call() throws Exception {
                Thread.sleep(500);
                int nivel = random.nextInt(101);
                System.out.println("🚗 Nivel de congestión: " + nivel + "%");
                return nivel > 70;
            }
        };

        Callable<Boolean> contaminacion = new Callable<Boolean>() {
            public Boolean call() throws Exception {
                Thread.sleep(600);
                int pm = 30 + random.nextInt(50);
```

Home End ( ) { } < > ' " ;

MeridianPrime.java

```
            System.out.println("🌫️ PM2.5: " + pm + " ug/m3");
            return pm > 50;
        }
    };

    Callable<Boolean> accidente = new Callable<Boolean>() {
        public Boolean call() throws Exception {
            Thread.sleep(800);
            String[] prioridades = {"Baja", "Media", "Alta"};
            String nivel = prioridades[random.nextInt(prioridades.length)];
            System.out.println("🚑 Accidente con prioridad: " + nivel);
            return "Alta".equals(nivel);
        }
    };

    Callable<Boolean> trenMaglev = new Callable<Boolean>() {
        public Boolean call() throws Exception {
            Thread.sleep(700);
            int retraso = random.nextInt(11);
            System.out.println("🚄 Retraso de tren maglev: " + retraso + " min");
            return retraso > 5;
        }
    };

    Callable<Boolean> semaforos = new Callable<Boolean>() {
        public Boolean call() throws Exception {
```

Home End ( ) { } < > ' " ;

MeridianPrime.java

```
            Thread.sleep(400);
            String[] estados = {"Verde", "Amarillo", "Rojo"};
            int contadorRojo = 0;
            for (int i = 0; i < 5; i++) {
                String estado = estados[random.nextInt(estados.length)];
                if ("Rojo".equals(estado)) {
                    contadorRojo++;
                } else {
                    contadorRojo = 0;
                }
                Thread.sleep(200);
            }
            System.out.println("🚦 Semáforo en rojo consecutivo: " + contadorRojo + " veces");
            return contadorRojo >= 3;
        }
    };

    // Lanzamos las tareas
    Future<Boolean> f1 = executor.submit(trafico);
    Future<Boolean> f2 = executor.submit(contaminacion);
    Future<Boolean> f3 = executor.submit(accidente);
    Future<Boolean> f4 = executor.submit(trenMaglev);
    Future<Boolean> f5 = executor.submit(semaforos);

    // Esperamos los resultados
    boolean critico1 = f1.get();
    boolean critico2 = f2.get();
```

Home End ( ) { } < > ' " ;

14:06

MeridianPrime.java

```
        boolean critico1 = f1.get();
        boolean critico2 = f2.get();
        boolean critico3 = f3.get();
        boolean critico4 = f4.get();
        boolean critico5 = f5.get();

        int totalCriticos = 0;
        if (critico1) totalCriticos++;
        if (critico2) totalCriticos++;
        if (critico3) totalCriticos++;
        if (critico4) totalCriticos++;
        if (critico5) totalCriticos++;

        System.out.println("\n📊 Resultado del análisis:");
        if (totalCriticos >= 3) {
            System.out.println("🚨 Alerta global: Se detectaron " + totalCriticos + " eventos críticos simultáneos.");
        } else if (totalCriticos > 0) {
            System.out.println("⚠️ Eventos críticos detectados: " + totalCriticos);
        } else {
            System.out.println("✅ Todos los sistemas operan dentro de parámetros normales.");
        }

        executor.shutdown();
    }
}
```

Home End ( ) { } < > ' " ;


14:06
Console
🛰️ Iniciando monitoreo de sistemas en Meridian Prime...
🚗 Nivel de congestión: 99%
🌫️ PM2.5: 38 ug/m3
🚈 Retraso de tren maglev: 10 min
🚑 Accidente con prioridad: Baja
🚦 Semáforo en rojo consecutivo: 1 veces
📊 Resultado del análisis:
⚠️ Eventos críticos detectados: 2
CTRL
ESC
HOME
END
PGUP
PGDN