```
13:50 # № •
                                                                                                                            ७ ⊝ ‡ 📦 🗎
                                                                                                                         :: × :
 ■ MonitorCPU.java
 1 import java.util.HashSet;
   import java.util.Scanner;
3
   import java.util.Set;
5
   class ConsumoCriticoException extends Exception {
6
       public ConsumoCriticoException(String mensaje) {
7
          super(mensaje);
8
9 }
10
11 public class MonitorCPU {
12
13
       public static void main(String[] args) {
14
          Scanner scanner = null;
15
          Set<Double> consumosRegistrados = new HashSet<Double>();
16
17
          try {
18
              scanner = new Scanner(System.in);
19
              System.out.println("Ingresa el consumo de CPU por servidor en porcentaje (ingresa 'fin' para terminar):");
20
21
              while (true) {
22
                  System.out.print("Consumo CPU: ");
23
                  String entrada = scanner.nextLine();
24
25
                  if ("fin".equalsIgnoreCase(entrada)) {
26
                      break;
                                                         { } < > ' "
                             Home End →I
                                                (
                                                   )
              >_
```

```
■ MonitorCPU.java
                                                                                                                          :: × ::
27
28
29
                  double consumo;
30
31
32
                      consumo = Double.parseDouble(entrada);
33
                  } catch (NumberFormatException e) {
                      System.out.println("[Error] Entrada no válida, debe ser un número.");
34
35
                      continue;
36
37
38
                  if (consumo < 0 || consumo > 100) {
39
                      System.out.println("[Error] El consumo debe estar entre 0 y 100.");
40
                      continue;
41
                  }
42
43
                  if (consumosRegistrados.contains(consumo)) {
44
                      System.out.println("[Aviso] Consumo duplicado, no será registrado.");
45
                      continue;
                  }
46
47
48
                  if (consumo > 95) {
                      throw new ConsumoCriticoException("Consumo crítico detectado: " + consumo + "%");
49
50
51
                  consumosRegistrados.add(consumo);
52
                                                    ) { } < > ' " ;
                         \vee Home End \rightarrowI (
               >_
```

७ ⊖ ‡ 📦 🗎

13:50 ╬ № •

```
■ MonitorCPU.java
                                                                                                                             [] X
                   if (consumo > 95) {
48
                       throw new ConsumoCriticoException("Consumo crítico detectado: " + consumo + "%");
49
50
51
52
                   consumosRegistrados.add(consumo);
                   System.out.println("Consumo " + consumo + "% registrado correctamente.");
53
               }
54
55
               System.out.println("\nRegistro finalizado. Consumos registrados:");
57
               for (Double c : consumosRegistrados) {
58
                   System.out.println("- " + c + "%");
59
60
61
           } catch (ConsumoCriticoException e) {
               System.out.println("[Alerta crítica] " + e.getMessage());
62
63
           } catch (Exception e) {
               System.out.println("[Error inesperado] " + e.getMessage());
64
65
           } finally {
               if (scanner \neq null) {
66
67
                   scanner.close();
                   System.out.println("Recurso Scanner cerrado.");
68
69
               }
70
           }
71
       }
72 }
73
                              Home End →I
                                                 ( )
                                                          {
               >_
```

७ ⊝ ‡ 📦 🗎

13:50 ╬ № •

E Console

Ingresa el consumo de CPU por servidor en porcentaje (ingresa 'fin' para terminar):
Consumo CPU: fin

Registro finalizado. Consumos registrados:
Recurso Scanner cerrado.

७ ⊝ 📦 🗎

13:50 ╬ № •