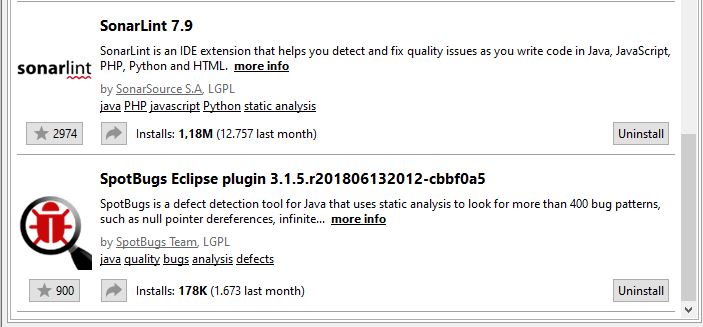
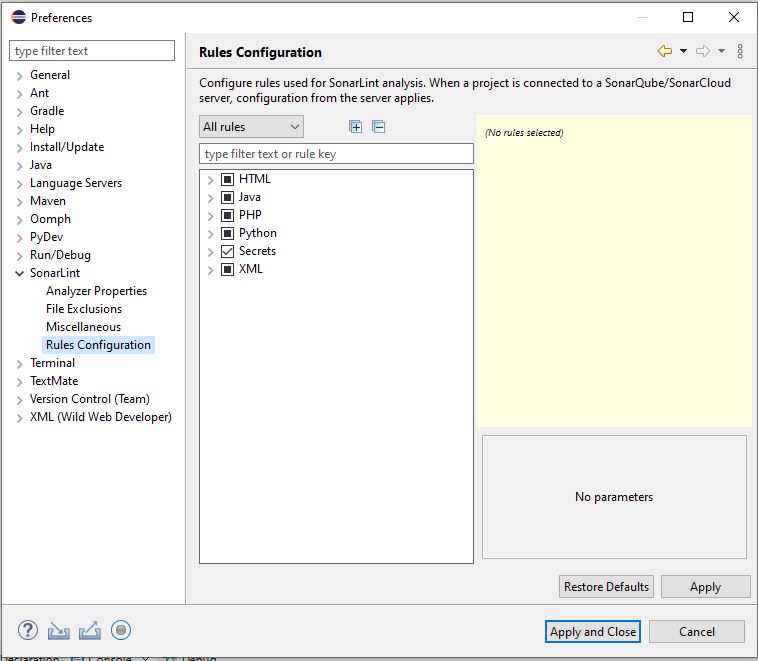
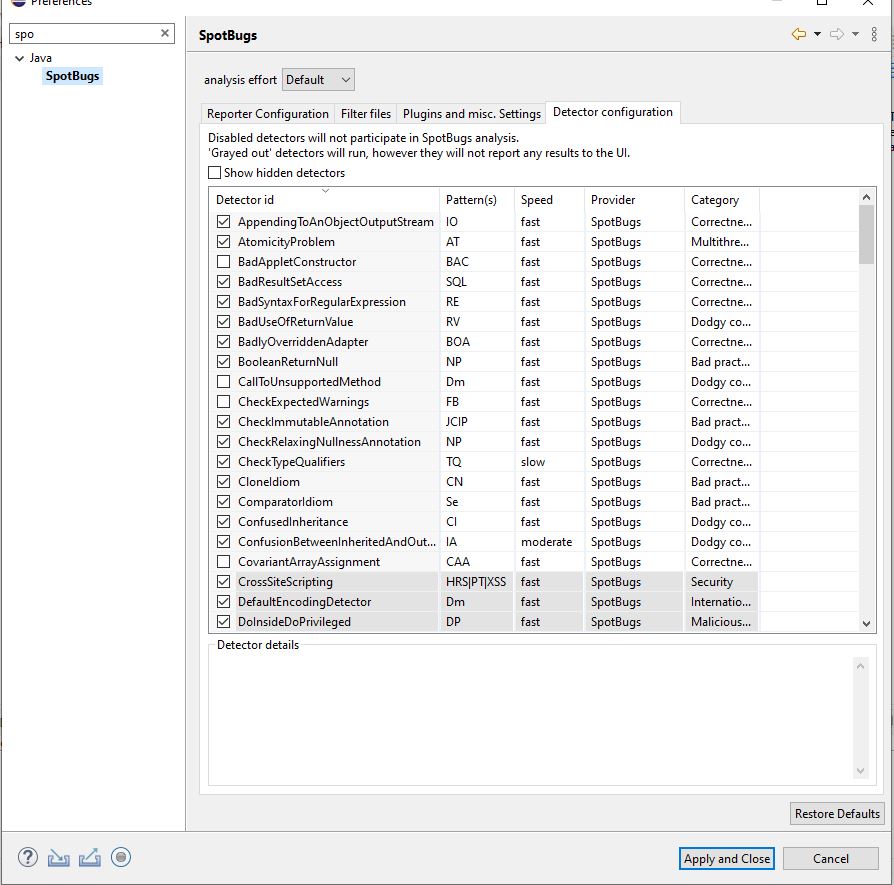
Boletín 2. Revisión de código:

A.

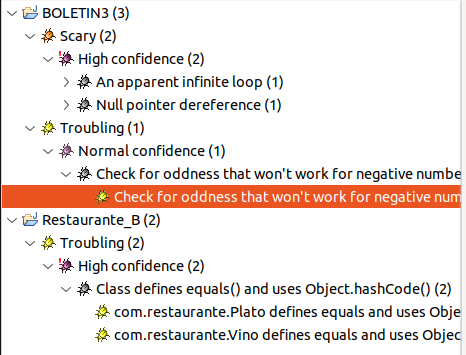
1.

2.





3.



4.

Errores analizados con Spotbug:

|  |  |
| --- | --- |
| Identificador.. | |
| Tipo de problema | Null pointer |
| Criticidad | Scary(5) |
| Descripción(código) | **if** (cadena!=**null** || !cadena.isEmpty()) { |
| Solución aplicada(código) | Crear una excepción para el null. |

|  |  |
| --- | --- |
| Identificador... | |
| Tipo de problema | com.restaurante.Plato defines equals and uses Object.hashCode() |
| Criticidad | Troubling(14) |
| Descripción(código) | HE\_EQUALS\_USE\_HASHCODE |
| Solución aplicada(código) | @Overrive |

|  |  |
| --- | --- |
| Identificador... | |
| Tipo de problema | Check for oddness that won't work for negative numbers in EJERCICIOS.ejercicio1.main(String[]) |
| Criticidad | Trobling(13) |
| Descripción(código) | IM\_BAD\_CHECK\_FOR\_ODD |
| Solución aplicada(código) | (i & 1) == 1 |

|  |  |
| --- | --- |
| Identificador... | |
| Tipo de problema | /home/estudiante/JAVA/ws/BOLETIN3/src/EJERCICIOS/ejercicio7.java:48 There is an apparent infinite loop in EJERCICIOS.ejercicio7.main(String[]) [Scary(8), High confidence] |
| Criticidad | High |
| Descripción(código) | Posibilidad de un bucle infinito. |
| Solución aplicada(código) | Incluir una excepcion de salidad |

|  |  |
| --- | --- |
| Identificador... | |
| Tipo de problema | Bug: com.restaurante.Vino defines equals and uses Object.hashCode() |
| Criticidad | Troubling (14) |
| Descripción(código) | HE\_EQUALS\_USE\_HASHCODE |
| Solución aplicada(código) | @Overrive |

Errores identificados con Sonarlint:

|  |  |
| --- | --- |
| Identificador... | |
| Tipo de problema | Unused local variables should be removed |
| Criticidad | Minor |
| Descripción(código) | public int numberOfMinutes(int hours) {  int seconds = 0; // seconds is never used  return hours \* 60;  } |
| Solución aplicada(código) | public int numberOfMinutes(int hours) {  return hours \* 60;  } |

|  |  |
| --- | --- |
| Identificador... | |
| Tipo de problema | Collapsible "if" statements should be merged |
| Criticidad | Major |
| Descripción(código) | if (file != null) {  if (file.isFile() || file.isDirectory()) {  /\* ... \*/  }  } |
| Solución aplicada(código) | if (file != null && isFileOrDirectory(file)) {  /\* ... \*/  } |

|  |  |
| --- | --- |
| Identificador... | |
| Tipo de problema | Unused assignments should be removed |
| Criticidad | Major |
| Descripción(código) | i = a + b; // Noncompliant; calculation result not used before value is overwritten  i = compute(); |
| Solución aplicada(código) | i = a + b;  i += compute(); |

|  |  |
| --- | --- |
| Identificador... | |
| Tipo de problema | Standard outputs should not be used directly to log anything |
| Criticidad | Major |
| Descripción(código) | System.out.println("My Message"); // Noncompliant |
| Solución aplicada(código) | logger.log("My Message"); |

|  |  |
| --- | --- |
| Identificador... | |
| Tipo de problema | String literals should not be duplicated (java:S1192) |
| Criticidad | Critical |
| Descripción(código) | ith the default threshold of 3: |
| Solución aplicada(código) | private static final String ACTION\_1 = "action1"; // Compliant  public void run() {  prepare(ACTION\_1); // Compliant  execute(ACTION\_1);  release(ACTION\_1);  } |

B.

Sonarlint: es una extensión IDE gratuita y open-source que identifica y te ayuda a solucionar los problemas de calidad y seguridad mientras escribes el código.

Es un gran aliado en la entrega de código sin errores. Gracias a esta herramienta inteligente los miembros de tu equipo podrán ver sus errores en tiempo real, comprenderlos y corregirlos rápidamente, de esta manera seguirán mejorando a lo largo del camino.

Se adapta perfectamente a tu IDE y es fácil de usar.

### Orientación clara

Descubre los problemas en su contexto con descripciones de reglas, una guía clara y ejemplos de código para que aprendas cómo solucionarlos.

### Flujo de datos de problemas

Resaltar todas las ubicaciones afectadas en tu código te permite tener una visión clara de los problemas que se deben abordar.

### Análisis rápido y preciso

El análisis de alta precisión sobre la marcha significa menos falsos positivos y resultados consistentes y confiables.

Spotbug: es un programa para detectar errores en código Java usando análisis estático.

 SpotBugs seleccionando el proyecto y la opción Find Bugs

En la pestaña Bug Explorer me aparecen los bugs categorizados por gravedadY pinchando en cada uno me remarcará el problema