P10- Análisis de pruebas: Cobertura

Revisión de las soluciones entregadas

Hemos revisado las entregas que habéis subido a Bitbucket de la práctica 10. A continuación, os indicamos los errores que hemos encontrado y añadimos también explicaciones sobre sus soluciones.

No obstante, si durante la realización o corrección de algún ejercicio os surgen otras dudas concretas, podéis enviarlas al foro si son de carácter general o enviar una tutoría si se trata de alguna cuestión particular de vuestra solución.

Errores detectados y soluciones

EJERCICIO 1:

- Cuando se indica que se quiere conseguir una cobertura del 100% de líneas y de condiciones, hay que considerar las dos a la vez, algunos lo habéis justificado por separado. Cuidado con esto. Siempre que se diga que se quiere una cobertura de nivelX y nivelY, nos estaremos refiriendo a que queremos conseguir ambos niveles a la vez.
- Para justificar el valor de CC, en el apartado A hay que usar las fórmulas que ya conocemos. A partir del apartado B), cuando se pide justificar los valores de cobertura del informe de JaCoCo, lógicamente tendréis que tener en cuenta la fórmula que usa JaCoCo.
- Los tests parametrizados no tienen por qué crearse en una nueva clase. No hay ningún problema en que la clase contenga tests parametrizados y sin parametrizar
- Tenéis tendencia a definir los datos de entrada, y salida como atributos de la clase que contiene los tests. NO HAGÁIS ESO! Los atributos de una clase no son variables globales. Es un error que hemos detectado no sólo en esta práctica. Recordad que no se trata solo de que el código funcione, sino de que esté correctamente implementado.
- En este primer ejercicio tenéis que familiarizaros con los informes de cobertura de JaCoCo y tenéis que tener claros qué significan los valores de los contadores, teniendo en cuenta que dichos informes se proporcionan en varios niveles.
- También hay que tener claro cómo hay que configurar el pom para poder generar los informes de cobertura de los tests unitarios.
- Para el ejercicio 1 hay que crear un proyecto Maven cuyo artifactID = "cobertura"

EJERCICIO 2:

- El ejercicio 2 se resuelve en el proyecto "cobertura" que hemos creado en el ejercicio1. En un paquete diferente.
- Tenéis que modificar correctamente el pom, para generar no sólo los informes de cobertura de los tests unitarios, sino también los informes de los tests de integración.
- También tenéis que familiarizaros con el uso de la goal del plugin JaCoCo para "chequear" diferentes niveles de cobertura. Los dos "chequeos" del apartado B) deben estar incluidos en la misma "rule".

EJERCICIO 3:

- El ejercicio 3 usa el proyecto multimódulo matriculación, y debe ser un proyecto totalmente independiente del proyecto "cobertura" de los ejercicios anteriores. Es decir, para esta práctica sólo tendremos dos proyectos maven: "cobertura" y "matriculacion" (independientemente de que matriculacion sea un proyecto multimódulo, y por lo tanto, contiene varios proyectos maven).

- Tenéis que justificar los valores obtenidos en los informes en los diferentes niveles indicados., es decir, tenéis que indicar qué fila y columna nos proporciona cada valor, y también de dónde sale dicho valor (cómo se calcula).

- También tenéis que familiarizaros con el uso de los "aggregate reports", y tener claro que un informe agregado de un determinado proyecto maven A, aglutina los informes de cobertura de las dependencias de dicho proyecto A.