ISFDyT N°70

Tecnicatura Superior en Análisis de Sistemas

Ingeniería de Software II

- **Trabajo Práctico 1 – Introducción a JUnit**
- Falabella Valentino
- 3° año
- **Nombre de la profesora:** Marina Caseres
- **Fecha de entrega:** 12/09/2025

1. Selección del programa a testear

- Cada estudiante deberá elegir un programa propio ya desarrollado previamente en la cursada, en el lenguaje JAVA.
- El programa debe tener al menos tres clases (ejemplo: Alumno, Curso, GestorAlumnos).

2. Pruebas unitarias por clase

- Crear una clase de prueba por cada clase del programa, ubicada en la carpeta test/.
- Cada clase de prueba debe cubrir los métodos principales y en su mayoría, de la clase original.
- El mínimo obligatorio es tres clases testeadas, pero si el programa tiene más, se deben testear todas.

3. Uso de anotaciones JUnit (Obligatorios):

- @BeforeAll y @AfterAll
- @BeforeEach y @AfterEach
- @Test
- @Disabled

Entre otras.

4. Uso de assertions básicas JUnit:

- assertEquals(expected, actual) – Verifica que dos valores sean iguales.
- assertNotEquals(unexpected, actual) – Verifica que dos valores no sean iguales.
- assertTrue(condition) – Verifica que la condición sea verdadera.
- assertFalse(condition) – Verifica que la condición sea falsa.
- assertNull(object) – Verifica que un objeto sea nulo.
- assertNotNull(object) – Verifica que un objeto no sea nulo.
- fail(message) – Falla el test explícitamente con un mensaje.
- Entre otras.

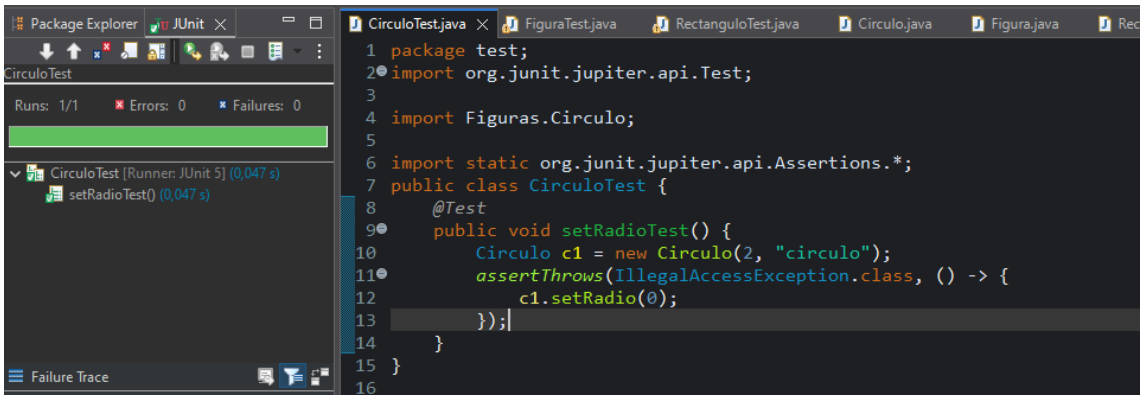
5. Pruebas con excepciones

- Incluir al menos tres pruebas por clase que verifique el lanzamiento de una excepción en condiciones inválidas (ejemplo: división por cero, índice fuera de rango, entrada nula, etc.).

- assertThrows(ExceptionClass.class, executable)

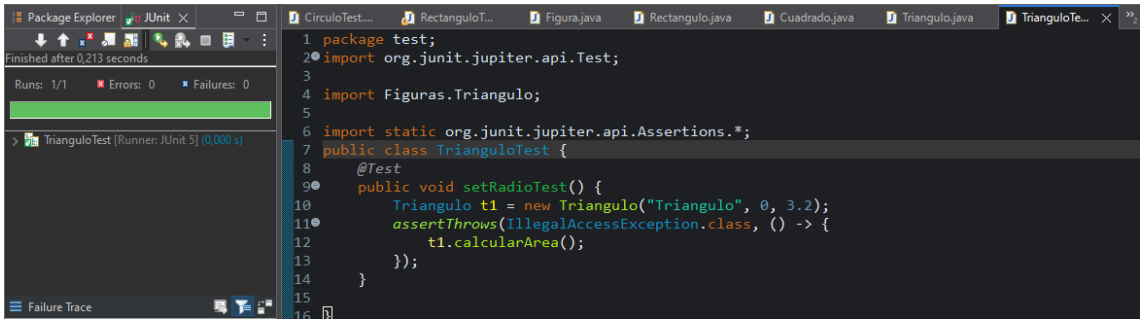
Pruebas

Clase Circulo:



La clase implementa el cálculo del área de un círculo a partir de su radio. En las pruebas se valida que el programa detecte correctamente el caso en que el radio ingresado sea **0**. Este valor no tiene sentido geométrico porque daría un área nula, por lo que el objetivo del test es confirmar que el código maneje esa condición y no permita un cálculo inválido.

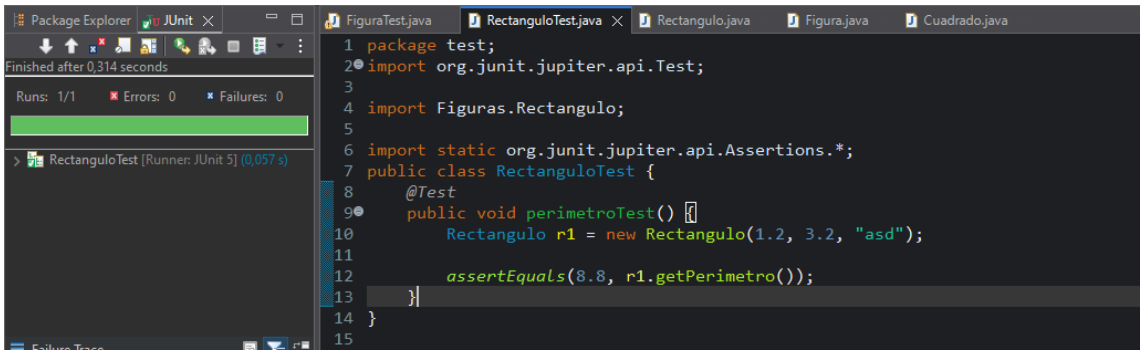
Clase Triangulo:



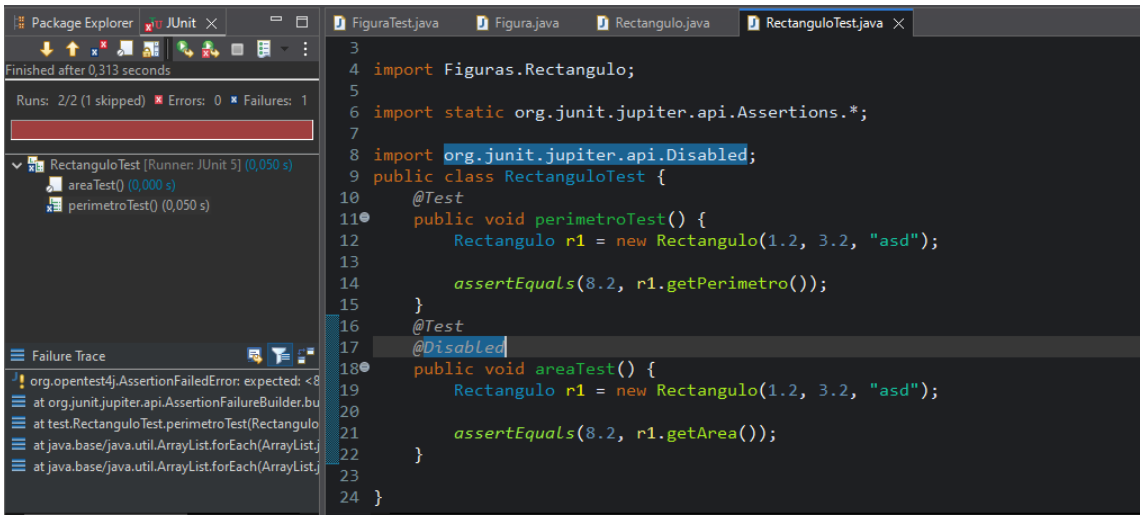
La clase calcula el área de un triángulo en base a su base y altura. En los tests se evalúa que, si alguno de los valores ingresados es **0**, el resultado no sea considerado válido. Esto se debe a que un triángulo con base o altura nula no puede existir. El objetivo de la prueba es garantizar que la clase detecte este error y devuelva el resultado esperado (ya sea excepción o área igual a 0).

Clase Rectángulo:

La clase obtiene el área de un rectángulo a partir de su base y altura. Se realizan pruebas comparando el resultado esperado con el calculado:



En un caso, la verificación es correcta porque el cálculo devuelve exactamente **8.8**, que es el valor esperado.

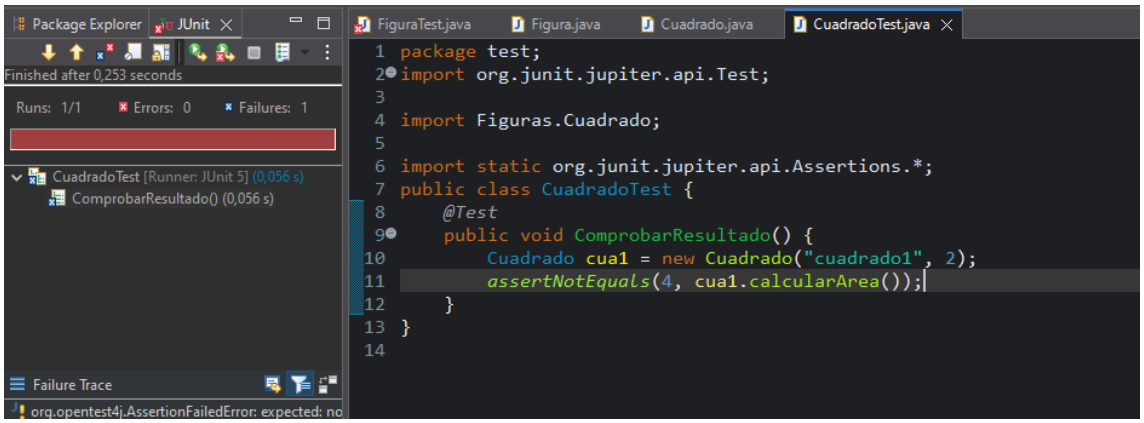


En este caso, se prueba contra un valor incorrecto (**8.2**) para confirmar que el método **no** lo acepte como válido.

Con esto se busca comprobar tanto el caso exitoso como el fallido.

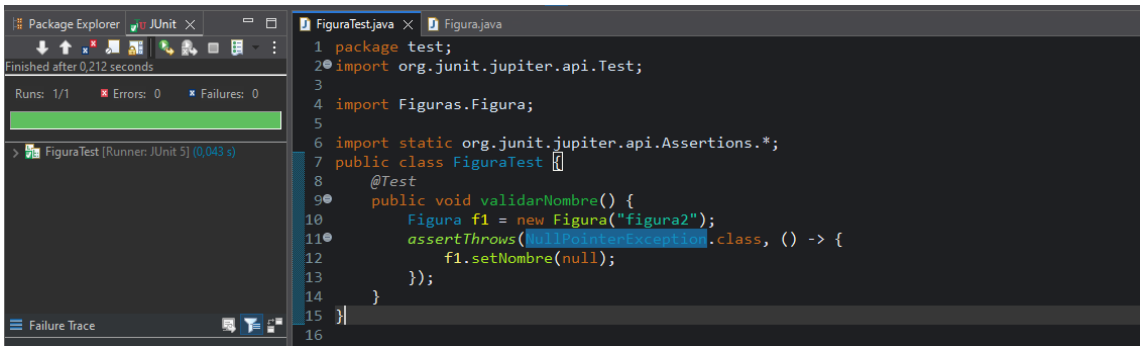
Además, se incluye un test con la anotación **@Disabled**, que sirve para desactivar temporalmente una prueba. En este ejemplo, se deja deshabilitado uno de los tests, mostrando cómo JUnit reporta “2/2 (1 skipped)”.

Clase Cuadrado:



La clase calcula el área de un cuadrado en base a la longitud de su lado. En la prueba se valida que el resultado **no** coincida con un valor específico (4). El test falla intencionalmente porque justamente el área sí resulta ser 4. El objetivo es mostrar cómo se comporta JUnit cuando una condición planteada no se cumple, ayudando a identificar posibles errores o supuestos incorrectos.

Clase Figura:



La clase se encarga de manejar características generales de una figura. El test verifica que el nombre de la figura no sea “Null”, lo cual asegura que siempre exista una referencia válida al identificador del objeto. De esta manera, se prueba que el código mantenga la consistencia de los datos y no permita trabajar con valores vacíos.

6. Gestión en GitHub (obligatorio)

- Crear un repositorio en GitHub con nombre: TP1_JUnit_ApellidoNombre.
- El repositorio debe incluir:
  - El programa original (todas las clases).
  - Las clases de prueba con JUnit.
  - Todos los archivos necesarios para compilar y ejecutar el proyecto.

7. El informe en PDF titulado:

Informe_TP1_JUnit_ApellidoNombre.pdf.

Informe en PDF (a incluir en el repositorio)

El informe debe contener:

- Portada con datos del estudiante, materia, profesora y fecha.
- Breve descripción del programa propio seleccionado.
- Listado de clases testeadas con explicación de los métodos cubiertos.
- Capturas de pantalla de la ejecución de las pruebas y los resultados obtenidos.
- Código fuente de las pruebas con comentarios explicativos sobre cada anotación.
- Reflexión final sobre la utilidad de JUnit y las pruebas unitarias en el desarrollo de software.