Ejercicios de JUnit

Índice

1. Multiplicación de matrices

Vamos a trabajar sobre el ejemplo de matrices visto en clase, y que también tenéis en la plantilla:

- Añadid a la clase *es.ua.jtech.jhd.sesion7.ejercicios.matrices.Matriz* de la plantilla un método *multiplicar(Matriz m)* que multiplique la matriz actual por la matriz *m*, y devuelva la matriz resultante. Se asume que las matrices son cuadradas y de igual dimensión M. Recordamos que en ese caso, el algoritmo de multiplicación C = A x B es:

```
para i = 1 hasta M
        para j = 1 hasta M
                para k = 1 hasta M
                        C[i][k] = C[i][k] + A[i][j] * B[j][k]
```

- A continuación, añadid un método *testMultiplicar()* en la clase *MatrizTest* del mismo paquete (en la carpeta *src* para los tests) que compruebe si la multiplicación que hacéis en *multiplicar(...)* es correcta. Podéis ayudaros de los otros métodos de prueba para construirlo: definid constantes, objetos de tipo *Matriz*, y comparar resultados.

  SUGERENCIA: probad a multiplicar la matriz identidad por sí misma, y comprobad que devuelva la matriz identidad.

- Ejecutad la clase *MatrizTest* para comprobar que todas las pruebas funcionan correctamente.
- En el método *suite()* de *es.ua.jtech.jhd.sesion7.ejercicios.matrices.MatrizSuite* añadid una nueva suite para probar el nuevo método *testMultiplicar()*.

```
public static Test suite()
{
        ...
        TestSuite suiteC = new TestSuite("suiteMultiplicacion");
        ...
        suiteC.addTest(new MatrizTest("testMultiplicar"));
        ...
        suite.addTest(suiteC);
}
```

- Ejecutad la clase *MatrizSuite* para comprobar que todas las pruebas funcionan correctamente.

NOTA: no es importante que la multiplicación de matrices funcione, pero sí que el método de prueba esté bien hecho.

2. Pruebas con varias clases (*)

La clase *es.ua.jtech.jhd.sesion7.ejercicios.geometria.MiVector* de la plantilla contiene un

array de enteros como campo. También tiene un método *maximo()* que devuelve el mayor elemento del array. Se pide:

- Construir la clase de prueba *es.ua.jtech.jhd.sesion7.ejercicios.matrices.MiVectorTest* (en la carpeta *src* para los tests), que chequee el método *maximo()* con el array {9, 3, 5, 7, 1} (el máximo sería 9).
- Ejecutad la prueba. Deberá fallar. ¿A qué se debe? Corregid el error en *MiVector* y repetid la prueba

La clase *es.ua.jtech.jhd.sesion7.ejercicios.geometria.Circulo* de la plantilla contiene los datos necesarios para un círculo en 2D, como son su coordenada X, coordenada Y y su radio. Se tienen los métodos *area()* y *longitud()* que devuelven, respectivamente, el área y longitud de arco del círculo. Se pide:

- Construir una clase *es.ua.jtech.jhd.sesion7.ejercicios.geometria.CirculoTest* en la carpeta de fuentes para tests, que chequee los métodos *area()* y *longitud()* de la clase *Circulo*.
- Construir una clase *es.ua.jtech.jhd.sesion7.ejercicios.geometria.GeometriaTest* en la carpeta de fuentes para tests, que construya una suite para ejecutar las pruebas de *MiVectorTest* y de *CirculoTest*.

  SUGERENCIA: podéis crear en *GeometriaTest* un método *suite* que se encargue de añadir una suite para *MiVectorTest* y otra para *CirculoTest*, y luego al ejecutar *GeometriaTest* se cargarán estas dos suites. También podéis, si preferís, utilizar el asistente de Eclipse para crear las suites de pruebas de estas dos clases. Al final, debería quedar algo más o menos como:

```
public static Test suite()
{
        TestSuite suite  = new TestSuite("suiteRaiz");
        suite.addTest(new TestSuite(MiVectorTest.class));
        ...
        return suite;
}
...
```

- Ejecutar *GeometriaTest* para testear todos los métodos de la suite.