**Escuela Colombiana de Ingeniería Julio Garavito**

**LABORATORIO NUMERO. 02**

**Nombres:** Camilo Murcia Espinosa, Jeisson Casallas

9/8/2023

**Conociendo el proyecto**

**1. El proyecto “TensorNP” contiene una construcción parcial del sistema. Revisen el directorio donde se encuentra el proyecto. Describan el contenido considerando los directorios y las extensiones de los archivos.**

* Un archivo README, el cual deberia contener toda la información del proyecto en Bluej, sin embargo se encuentra vacio.
* Package.bluej, que es el ejecutable en donde podemos ver nuestro proyecto en Bluej

**Por cada una de las clases definidas en el Proyecto contiene lo siguiente:**

* Un archivo con extensión .class, que es el archive compilado de la clase el cual es el que podemos evidenciar al ejecutar el BlueJ.
* Un archive con extensión .ctxt, que contiene información adicional, como la documentación y los comentarios que se hacen acerca de cada clase y sus metodos.

* Un archive con extensión .java, que es el que contiene el codigo fuente de la clase.

**2. Exploren el proyecto en BlueJ**

**¿Cuántas clases tiene?**

Tiene 3 clases, que son NPTensor, Tensor y TensorTest

**¿Cuál es la relación entre ellas?**

Todas las clases se relacionan gracias a la clase de Tensor, que une a NPTensor y TensorTest

**¿Cuál es la clase principal de la aplicación? ¿Cómo la reconocen?**

La clase principal es NPTensor, ya que es la clase que ofrece los servicios de la aplicación. En este caso la clase que permite almacenar y operar tensores.

**¿Cuáles son las clases “diferentes”? ¿Cuál es su propósito?**

La clase diferente tiene el nombre TensorTest, su propósito es realizar pruebas sobre la clase Tensor, para verificar que esta cumple con todos sus requisitos, y realiza su trabajo de manera correcta.

**Para las siguientes dos preguntas sólo consideren las clases “normales”:**

**3. Generen y revisen la documentación del proyecto: ¿está completa la documentación de cada clase? (Detallen el estado de documentación de cada clase: encabezado y métodos)**

No, la documentacion esta incompleta, ya que solo se tiene un pequeño comentario sobre el constructor, pero no se especifican los parametros que entran en este, por otra parte, los demas metodos no tienen ninguna documentacion.

La clase de NPTensor esta igualmente incompleta y no cuenta con toda la documentacion correspondiente, ni en su encabezado ni en los metodos.

La clase de TensorTest tampoco esta documentada de forma correcta, por lo que cuando se genera la documentacion, no obtenemos los resultados que se esperarian.

**4. Revisen las fuentes del proyecto, ¿en qué estado está cada clase? (Detallen el estado de las fuentes considerando dos dimensiones: la primera, atributos y métodos, y la segunda, código, documentación y comentarios)**

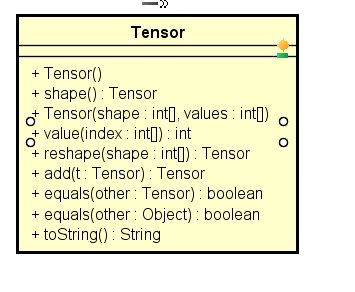
* **Tensor:** Contiene los metodos necesarios para ejecutar el archivo y compile, pero su codigo en muchos casos, tiene metodos “vacios” o sin codigo.
* **NPTensor:** Contiene los metodos necesarios para ejecutar el archivo y compile, pero su codigo en muchos casos, tiene metodos “vacios” o sin codigo.
* **TensorTest:** Contiene los metodos necesarios para ejecutar el archivo y compile, pero su codigo en muchos casos, tiene metodos “vacios” o sin codigo.

**¿Qué son el código, la documentación y los comentarios?**

* El código, la documentación y los comentarios son todos importantes para la creación de programas informáticos. El código es lo que hace que un programa informático funcione, la documentación explica cómo usar el programa informático y los comentarios explican lo que hace el código.

**Ingenieria Reversa**

**1. Complete el diagrama de clases correspondiente al proyecto (No incluya la clase de pruebas)**



**2. ¿Cuál nuevo contenedor está definido? Consulte la especificación y el API Java 1¿Qué diferencias hay entre el nuevo contenedor, el ArrayList y el vector [] que conocemos?**

Los contenedores que se utilizan en el laboratorio son dos: HashMap y Array.

Las diferencias que hay entre ellos, es que HashMap almacena una pareja llave-valor, en una estructura de datos tipo HashTable, en donde además se permiten llaves nulas; el Array por otro lado, requiere saber su tamaño al momento de su inicialización, y almacena sus datos de manera secuencial en memoria

**Conociendo Pruebas en BlueJ**

Para poder cumplir con la prácticas XP vamos a aprender a realizar las pruebas de unidad usando las herramientas apropiadas. Para eso consideraremos implementaremos algunos métodos en la clase TensorTest

**1. Revisen el código de la clase TensorTest. ¿cuáles etiquetas tiene (componentes con símbolo @)?**

Se encuentran las etiquetas @test, @BeforeClass y @before

**¿cuántos métodos tiene?**

Hay 6 metodos en total(beforeClass(), before(), shouldCreateTensor(), shouldKnowWhenTwoTensorAreEquals(), shouldRepresentATensorAsAString() y shouldAdd())

**¿cuantos métodos son de prueba? ¿cómo los reconocen?**

4 Metodos tiene la etiqueta @test, que son shouldCreateTensor(), shouldKnowWhenTwoTensorAreEquals(), shouldRepresentATensorAsAString() y shouldAdd(), y se reconocen por su etiqueta

**2. Ejecuten los tests de la clase TensorTest. (click derecho sobre la clase, Test All) ¿cuántas pruebas se ejecutan? ¿cuántas pasan? ¿por qué?**

* Se ejecuta unicamente la primera prueba, beforeClass() el cual arroja error debido a que debe ser estatico, por lo que hasta no solucionar esto, no se van a ejecutar las demas y no pasa ninguna.

**3. Estudie las etiquetas encontradas en 1. Expliquen en sus palabras su significado.**

@test es una etiqueta perteneciente al framework Junit, y es la encargada de decirle al compilador que el método ‘public void’ bajo ella puede ser ejecutado como un caso de prueba. Si el método no arroja ninguna excepción, Junit asumirá que el caso de prueba fue exitoso.

El codigo que se pone con @before es ejecutado antes de cada test, mientras que el codigo con @BeforeClass se corre una vez, antes de todas las otras pruebas del @test, por ejemplo si tenemos una clase con 10 tests, el codigo con etiqueta @Before se ejecutara 10 veces, pero el codigo con @BeforeClass se ejecutara una unica vez.

**4. Estudie los métodos assertTrue, assertFalse, assertEquals, assertArrayEquals, assertNull y fail de la clase assert del API JUnit. Explique en sus palabras que hace cada uno de ellos.**

assertTrue: Toma como argumento una condición booleana, y se asegura de que esta sea verdadera, si no es así, arroja un AssertionError.

assertFalse: Toma como argumento una condición booleana, y se asegura de que esta sea falsa, si no es así, arroja un AssertionError.

assertEquals: su argumento son dos variables del mismo tipo, y esta función se encarga de verificar si esas dos variables tienen el mismo valor, para números de punto flotantes (float, double), se puede proporcionar un delta, es decir, el error máximo que puede ocurrir entre los dos valores.

assertArrayEquals: Se utiliza para verificar que dos arrays son iguales, y se asegura si es falsa o verdadera, si no es asi arroja un AssertionError.

assertNull: Toma como argumento un objeto, cuyo valores debe ser null, si no es así, arroja un AssertionError.

fail: Falla un test, con un mensaje opcional.

**5. Investiguen la diferencia que entre un fallo y un error en Junit. Escriba código, usando los métodos del punto 4., para lograr que los siguientes tres casos de prueba se comporten como lo prometen shouldPass, shouldFail, shouldErr.**

La diferencia entre un fallo y un error en Junit, es que el primero significa que el código no cumple con los criterios de la prueba de aceptación, es decir las aserciones definidas en las pruebas no pasan en su totalidad, el segundo se da cuando ocurre un error en el tiempo de ejecución, como por ejemplo dividir entre cero o acceder a una variable nula.

En JUnit, un fallo es una condición que se esperaba que se cumpliera, pero no lo hizo. Un error es una condición que no se esperaba que se produjera, pero sí lo hizo.

**Practicando Pruebas en BlueJ**

1. **Determinen los atributos de la clase *Tensor.* Justifique la selección.**

int [] shape un arreglo que contiene las dimensiones del arreglo que se le pase como parámetro.

int value contiene el número de elementos que le ingresan al arreglo

int[] values

1. **Determinen el invariante de la clase Tensor. Justifique la decisión.**
2. **3. Implementen únicamente los métodos de Tensor necesarios para pasar todas las pruebas definidas. ¿Cuáles métodos implementaron?**
3. **4. Capturen los resultados de las pruebas de unidad.**