**Índice**

[Enunciado 2](#_Toc371330888)

[Desarrollo 3](#_Toc371330889)

[Clases 4](#_Toc371330890)

[Test 4](#_Toc371330891)

[TestCase 4](#_Toc371330892)

[TestSuite 4](#_Toc371330893)

[Assertion 4](#_Toc371330894)

[TestResult 4](#_Toc371330895)

[FailureExcepcion 4](#_Toc371330896)

[Diagrama de clases 5](#_Toc371330897)

Enunciado

Desarrollo

Para llegar a la solución se decidió usar el patrón *Composite*. Test es la clase de la cual heredan el componente y el contenedor. El componente es un test específico, mientras que el contenedor es un grupo de tests. Es el único momento en el que se usa la herencia. Si bien no se utiliza ahora, esto permitirá agrupar distintos tipos de Tests luego. Por ejemplo si quisieran agruparse tests para un producto, podrían agruparse dentro de un contenedor que tenga productos. Luego este formará parte de otro contenedor general que tendrá éste y otros Tests, ya sean simples o compuestos.

Clases

Test

Esta clase fue creada para poder aplicar el patrón Composite.

TestCase

Clase que representa cada “test individual”, cada prueba en particular. Los Assert se deben utilizarse en esta clase.

TestSuite

Clase que contiene a los tests existentes. Además tiene un método para correrlos todos.

Assertion

Es la clase encargada de devolver el resultado de la comparación entre el/los operando/s recibido/s. El resultado de la operación lo arroja mediante excepciones definidas por nosotros.

TestResult

Clase que es contenedor del resultado de las pruebas. Se pasará un TestResult por parámetro al TestSuite, quien lo irá pasando a todos los tests que contiene para que lo completen. Posee 3 listas, en las cuales se guardarán los tests pasados (OK), fallados y que dieron error, según corresponda.

FailureExcepcion

Excepción utilizada para indicar que la comparación de los operandos dio un resultado negativo.

Diagrama de clases