ESCUELA COLOMBIANA DE INGENIERÍA
PROGRAMACIÓN ORIENTADA A OBJETOS
Septiembre 2017
Laboratorio 2/6

OBJETIVOS

Desarrollar competencias básicas para:

1. Desarrollar una aplicación aplicando BDD y MDD.
2. Realizar diseños (directa e inversa) utilizando una herramienta de modelado (astah)
3. Manejar pruebas de unidad usando un *framework (junit)*
4. Apropiar nuevas clases consultando sus especificaciones (API java)
5. Experimentar las prácticas XP : **Coding** Code the unit test first. **Testing** All code must have unit tests.

ENTREGA

- ✓ Incluyan en un archivo .zip los archivos correspondientes al laboratorio. El nombre debe ser los dos apellidos de los miembros del equipo ordenados alfabéticamente.
- ✓ En el foro de entrega deben indicar el estado de avance de su laboratorio y los problemas pendientes por resolver.
- ✓ Deben publicar el avance al final de la sesión y la versión definitiva en la fecha indicada en los espacios preparados para tal fin

CONTEXTO

Objetivo

En este laboratorio es desarrollar una calculadora de pila para vectores.
Estas calculadoras funcionan guardando sus operandos en una pila. Las operaciones las realizan sacando los operandos necesarios de la pila y adicionando el resultado a la misma.

Conociendo el proyecto [En lab02.doc]

1. El proyecto BlueJ "`Calculadora`" contiene una construcción parcial del sistema. Revisen el directorio donde se encuentra el proyecto. Describan el contenido considerando los apellidos de los archivos?.
2. Exploren el proyecto en BlueJ
   - ¿Cuántas clases tiene? ¿Cuál es la relación entre ellas?
   - ¿Cuál es la clase principal? ¿Cómo la reconocen?
   - ¿Cuáles son las clases "diferentes"? ¿Cuál es su propósito?

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? (Detalle el estado de documentación de cada clase: encabezado y métodos)
4. Revisen el código del proyecto, ¿en qué estado está cada clase? (Detalle el estado de codificación)

Ingeniería reversa [En calculadora.asta]

MDD MODEL DRIVEN DEVELOPMENT

Genere el diagrama de clases correspondiente a `Calculadora` con todos sus elementos.
(No incluya la clase de pruebas)

Conociendo Pruebas en BlueJ [En lab02.doc *.java]

De TDD → BDD (TEST → BEHAVIOUR DRIVEN DEVELOPMENT)

Para poder cumplir con la prácticas XP vamos a aprender a realizar las pruebas de unidad usando las herramientas apropiadas. Para eso consideraremos como ejemplo la clase `Angulo`

1. Revisen el código de la clase `Angulo .` ¿Qué condiciones se imponen para la implementación? ¿Cuáles son los diferentes tipos de representaciones? ¿Cómo podemos hacer conversiones entre ellas? ¿qué son objetos inmutables? ¿qué ventajas tienen?
2. Revisen el código de la clase `AnguloTest`. ¿Cuántas pruebas se tienen? ¿Cuáles están implementadas?
3. Ejecuten los tests de la clase `AnguloTest`. (`click derecho sobre la clase, Test All`) ¿cuántos tests se ejecutan? ¿cuántos pasan las pruebas? ¿por qué podrían pasar las pruebas?
4. Genere un caso de prueba que pase y otro que genere un error. ¿qué diferencia existe entre un error y un fallo?
5. Esto estudie los métodos `assertTrue, assertEquals y fail` de la clase `Assert` del API JUnit [1]. Explique en sus palabras que hace cada uno de ellos.
6. Adicionen ejemplares al caso de prueba `deberiaPoderExpresarUnAnguloComoCadena`
7. Completen los casos de prueba que no están implementados.
8. Para cada uno de los casos de prueba, indique los métodos que deben ser implementados para pasar la prueba e implémentelos.
9. Desarrolle uno de los métodos de la clase `Angulo` no implementados con la propuesta TDD_BDD:
   a. Implemente mínimo dos casos de prueba significativos
   b. Implemente el método para que pase las pruebas

Desarrollando

BDD - MDD

[En lab02.doc, calculadora.asta, *.java]

Para desarrollar esta aplicación vamos a considerar los siguientes ciclos de desarrollo.

Ciclo 1 : Adicionar , verificar y consultar
Ciclo 2 : Eliminar y duplicar
Ciclo 3 : Sumar y restar
Ciclo 4 : Multiplicar

En cada mini-ciclo deben realizar los pasos definidos a continuación.

1. **Definir los métodos base de Calculadora correspondientes al ciclo actual**
2. **Generar y programar los casos de prueba (piense en todos los deberia y en todos los noDeberia)**
3. **Diseñar los métodos (use diagramas de secuencia)**
4. **Generar y programar los casos de prueba de los métodos de la solución (piense en todos los debería y en todos los noDebería)**
5. **Escribir el código correspondiente (no olvide la documentación)**
6. **Ejecutar las pruebas de unidad (vuelva a 3 (a veces a 2). si no están en verde)**

1(http://junit.org/javadoc/latest/)

Completen la siguiente tabla indicando el número de ciclo y los métodos asociados de cada clase.

Ciclo	Calculadora	CalculadoraTest	Vector	VectorTest	Angulo	AnguloTest

RETROSPECTIVA

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/Hombre)
2. ¿Cuál es el estado actual del laboratorio? ¿Por qué?
3. Considerando las prácticas XP del laboratorio. ¿cuál fue la más útil? ¿por qué?
4. ¿Cuál consideran fue el mayor logro? ¿Por qué?
5. ¿Cuál consideran que fue el mayor problema técnico? ¿Qué hicieron para resolverlo?
6. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados?