ESCUELA COLOMBIANA DE INGENIERÏA
PROGRAMACIÓN ORIENTADA A OBJETOS
Agosto 2015
Laboratorio 2/6

OBJETIVOS

Desarrollar competencias básicas para:

1. Desarrollar una aplicación aplicando BDD y MDD.
2. Generar los diagramas de casos de uso y de clases dado el código de una aplicación.
3. Utilizar astah para realizar diseños.
4. Utilizar junit un framework para manejo de pruebas de unidad
5. Experimentar la práctica XP : **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 afabéticamente.
- ✓ 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 fín

CONTEXTO

Objetivo

En este laboratorio es desarrollar una calculadora de pila de polinómios.

Las calculadoras de pila funcionan guardando sus operandos en una pila. Las operaciones las realizan sacando los dos operando de la pila y adicionando el resultado a la misma.

Conociendo el proyecto [En lab02.doc]

1. El proyecto BlueJ "`CalculadoraPolinomio`" contiene una construcción parcial del sistema. Revisen el directorio donde se encuentra el proyecto. ¿Describa el contenido considerando los apellidos de los archivos?
2. Exploren en BlueJ
   ¿Cuántas clases tiene? ¿Cuál es la relación entre ellas?
   ¿Cuál es la clase principal? ¿Cómo se reconoce?
   ¿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. Editen el código del proyecto, ¿en qué estado está cada clase?

Ingeniería reversa [En calculadora.asta]

MDD MODEL DRIVEN DEVELOPMENT

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

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

TDD TEST 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 `Fraccionario`

1. Revisen el código de la clase `Fraccionario.` ¿Qué condiciones se imponen para la implemantación? ¿Cuándo una fracción es irreductible? ¿Cómo podemos convertir un fraccionario a su versión irreductible? ¿qué son objetos inmutables? ¿qué ventajas tienen?
2. Revisen el código de la clase `FraccionarioTest`. ¿Cuántos "tests" se tienen? ¿Cuáles están implementados?
3. Ejecuten los tests de la clase `FraccionarioTest`. (`click derecho sobre la clase, Test All`) ¿cuántos tests se ejecutan? ¿cuántos pasan las pruebas? ¿por qué pasan las pruebas? Ajusten las pruebas.
4. Revisen en detalle el código de los métodos de prueba. Para esto estudie los métodos `assertTrue, assertEquals y fail` de la clase `Assert` del API JUnit [1]. ¿Qué métodos debe implementar para que la clase pase TODOS los métodos de prueba?
5. Realice los cambios necesarios para que la clase pase TODOS los métodos de prueba.

Desarrollando

BDD - MDD

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

Para desarrollar esta aplicación vamos a considerar cuatro ciclos de desarrollo.

Ciclo 1 : Crear una calculadora, borrar los operandos, adicionar un operando y consultar
Ciclo 2 : Sumar y restar
Ciclo 3 : Multiplicar y dividir
Ciclo 4 : Integrar y derivar

En cada minicico debe realizar los pasos definidos a continuación.

1. **Defina los métodos base de Calculadora correspondientes al ciclo actual**
2. **Genere y programe los casos de prueba (piense en todos los debería y en todos los noDebería)**
3. **Diseñe los métodos (use diagramas de secuencia)**
4. **Genere y programe los casos de prueba de los métodos de la solución (piense en todos los debería y en todos los noDebería)**
5. **Escriba el código correspondiente (no olvide la documentación)**
6. **Ejecuten las pruebas de unidad (vuelva a 3 (a veces a 2). si no están en verde)**

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

Ciclo	Calculadora	CalculadoraTest	Polinomio	PolinomioTest	Fraccionario	FraccionarioTest

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

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 la práctica XP All code must have unit tests. ¿por qué consideran que es importante?
4. ¿Cuál consideran fue su mayor logro? ¿Por qué? ¿Cuál consideran que fue su mayor problema? ¿Qué hicieron para resolverlo?
5. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados?