Hoja de trabajo No. 4

Realizar: Programa: CALCULADORA para evaluar expresiones postfix.
Realizarse: en parejas**.**

Objetivos:
a. Utilización de genéricos.
b. Diseño del ADT para pilas (stack), debe tener: 1. Interfaz, 2. Clase Abstracta, 3. Clases de implementación.
c. Diseño del ADT para listas, con: 1.Interfaz, 2. Clase Abstracta, 3. Clases de implementación.
d. Uso de patrones de diseño: Factory y Singlenton.
e. Control de versiones del programa.
f. Emplear JUnit para casos de prueba.

Programa a realizar:
Su programa debe realizar el cálculo de una expresión postfix, tal como se realizó en la hoja de trabajo no. 2. Debe leer esa expresión de un archivo de texto llamado **datos.txt.**

Por ejemplo, el cálculo de la expresión postfix: `1 2 + 4 * 3 +`

La expresión es evaluada de izquierda a derecha utilizando una pila: el resultado es 15

NOTA: esta hoja implementa varios patrones de diseño. Su programa debe solicitar al usuario que indique la implementación deseada para el stack (arrayList, vector, lista). Si se desea usar la implementación de stack basada en listas, entonces también debe solicitar al usuario que indique la implementación de listas a emplear (simplemente encadenada, doblemente encadenada, circular).

Tareas:
a. Diagrama UML de clases. NO utilice ingeniería reversa. Muestre la relación entre las clases que usa para la pila y las utilizadas para la lista.

b. Construir la interfaz de la Pila, su clase abstracta y las clases de implementación con: 1. ArrayList, 2. Vector, 3. Lista. Debe utilizar genéricos.

c. Construir la interfaz de la Lista, su clase abstracta y las clases de implementación con: 1. Simplemente encadenadas, 2. Doblemente encadenadas, 3. Lista circular. Debe utilizar genéricos.
NOTA: no es necesario que implemente todos los métodos de la lista. Implemente solo aquellos que sean necesarios para soportar la operación de una pila.

d. Su programa principal (realizado en la hoja no. 2) debe usar:
Patron de diseño Factory: para seleccionar la implementación de la pila a utilizar. Favor notar que si selecciona una implementación basada en listas, deberá utilizar nuevamente ese patrón para indicar la implementación de lista a utilizar.
Patron de diseño Singlenton: para asegurarse que existe solo una instancia de su clase Calculadora. Esta es independiente de la GUI que usted utilice.

e. Debe dejar evidencia de todo el desarrollo en el repositorio de Git o sistema similar para control de versiones. Indicar como acceder a su repositorio y si es necesario, agregar a su catedrático y auxiliar para que tengan acceso al mismo.

f. Incluya pruebas unitarias con JUnit para las clases de implementación de pila y la clase lista.

Debe subir a Blackboard todos los productos elaborados y los enlaces a su repositorio de Git (o similar).

Calificación: deben existir los diagramas de Clases para que sea calificado su programa.

Aspecto	Puntos
Documentación generada con Javadoc, tiene precondiciones y postcondiciones en los métodos de los ADT empleados	5
Uso del repositorio: existen más de tres versiones guardadas, la última versión es igual a la colocada en el Blackboard.	5
Diagrama de clases: muestran la abstracción y encapsulación de las operaciones.	15
Utilización de los patrones de diseño Factory y Singlenton	20
ADT pila, con uso de genéricos.	20
ADT lista, con uso de genéricos	20
Funcionamiento del programa	15
Pruebas JUnit para las operaciones de la pila.	20
TOTAL:	120