CC2003 – Algoritmos y Estructura de Datos

Hoja de trabajo
No. 2

MANUEL ARMANDO ULIN PEREZ

CRISTA BOTSCHI DIEGUEZ

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

Objetivos:
a. Utilización de genéricos.
b. Diseño del ADT para pilas (stack).
c. Implementación de pilas con un vector de tamaño variable. (Clase VECTOR)
d. Control de versiones del programa.
e. Emplear JUnit para casos de prueba.

Programa a realizar:
Su programa debe leer de un archivo de texto, una expresión en formato Postfix y producir el resultado de la misma. El archivo de texto se llama **datos.txt** y será proporcionado por su auxiliar al correr el programa. En cada línea del archivo de texto vendrá una expresión en notación postfix, semejante a:

```
1 2 + 4 * 3 +
```

NOTA: Para simplificar su programa, cada operando es entero y de un solo dígito, y están separados de los operadores por un espacio en blanco. Los operadores son los aritméticos: + (suma), - (resta), * (multiplicación), / (división). La expresión v endrá en una sola línea del archivo de texto.

Adjunto un ejemplo tomado de la wikipedia:

Por ejemplo, el cálculo: ((1 + 2) * 4) + 3, puede ser anotado como en notación postfix con la ventaja de no necesitar paréntesis. La expresión postfix que vendrá en el archivo de texto para este ejemplo es:

```
1 2 + 4 * 3 +       esta es la expresión que vendrá en archivo de texto

Resultado: 15         esta es la salida que produce su programa.
```

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

1. **Operando:** cuandose encuentra un dígito (es un operando) entonces hacer push
2. **Operador**: cuandose encuentra un signo de operación, realizar el pop para tener dos operandos y evaluar el v alor cuando se encuentra una operación, y push el resultado. Cuidado, se debe preservar el orden de los operandos (algunas operaciones no son conmutativas), por lo que se opera así:
   operandoB = pop( )
   operandoA = pop( )
   resultado = operandoA operador operandoB
   push(resultado)

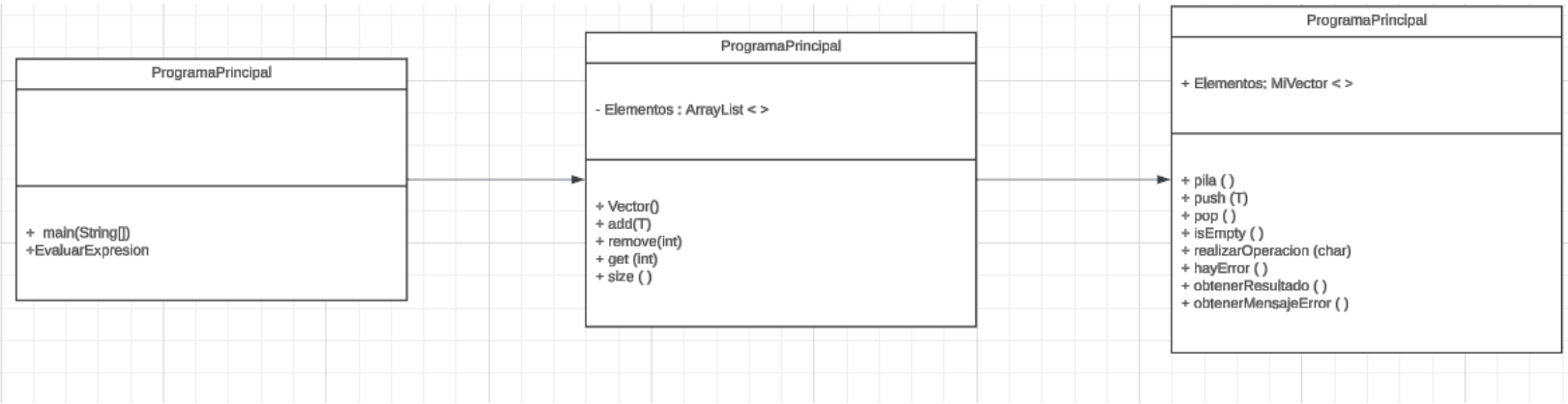
De la siguiente manera (la Pila se muestra después de que la operación se

haya llevado a cabo): ENTRADA OPERACION PILA (va

creciendo hacia la derecha)

1	push operando	1
2	push operando	1, 2
+	Sumar: pop, pop y push del resultado	3
4	push operando	3, 4
*	Multiplicar: pop, pop y push del resultado	12
3	push operando	12, 3
+	Sumar: pop, pop y push del resultado	15

A continuación se presenta el diagrama UML del programa:



Otro ejemplo de expresión:
6 2 3 + *
¿Qué resultado debe dar?

Tareas:

a. Diagrama UML de clases y de **secuencia**. NO utilice ingeniería reversa. En el diagrama de secuencia modele la comunicación entre objetos o clases para lograr la evaluación de la expresión Postfix. Use notación UML 2
b. Construir la interfaz de la Pila y la clase de implementación con un **vector de tamaño variable (VECTOR)**. Debe utilizar genéricos.
c. Su programa principal debe permitir cambiar la clase pila, sin que se vea afectado en su funcionamiento. Es decir, se puede usar la implementación basada en ArrayList (que se vió en el aula) o la nueva implementación que se está desarrollando en esta hoja, basada en Vector.
d. Debe crear un ADT para la calculadora paraexpresiones POSTFIX. Este ADT debe permitir que su programa principal funcione con la calculadora desarrollada por CUALQUIER grupo y la calcu ladora qu e desarrolle su grupo debe funcionar en CUALQUIER programa principal. Considerar quese pueden t ener erros numéricos como división entre cero, que hay un carácter que no puede ser int erpretado como operando o que hay insuficiente cantidad de operandos para realizar unaoperación.
e. Debe dejar evidencia de todo el desarrollo en el repositorio de git (o el sistema de control de versiones que usted utilice) .
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 JUnitpara la clase pila y para la calculadora.

Debe subir a Canvas todos los productos elaborados en los incisos a, b, c, e, f y los enlaces a su repositorio git.

Calificación: debenexistir los diagramas de Clases y Secuencia para que sea calificado su programa. Su programa principal debe poder usar la Calculadora elaboradapor cualquier persona.

Aspecto	Puntos
Estilo de codificación: comentarios, identación, nombres de variables significativas.	5
Documentación generada con Javadoc, tiene precondiciones y postcondiciones en los métodos del ADT Calculadora.	5
Uso del repositorio: existen más de tres versiones guardadas, la última versión es igual a la colocada en el Canvas	10
Diagrama de clases: muestran la abstracción y encapsulación de las operaciones.	10
Diagrama de secuencia muestra la evaluación de la expresión postfix	10
ADT pila, con uso de genéricos.	10
ADT Calculadora. Debecorrer en cualquier programa principal desarrollado por los otros grupos.	20
Su programa principal debe correr con el ADT Calculadora desarrollado por cualquiera de los otros grupos.	20
Pruebas JUnitpara las operaciones de la pila y de la calculadora.	10
TOTAL:	100