Ing. Adrián Gómez

Homework # 1 Strings

1.- Agregar comas a strings numéricos (2 puntos)

En algunos países es normal que números largos escritos sobre un papel sean separados por comas en grupos de tres. Por ejemplo un millón es escrito de la siguiente forma: 1,000,000.
Para que sea más fácil para los otros programadores mostrar los números de esta manera, implementa la solución:

class : StringUtil
method: String addCommasToNumericString(String digits)
type: static

2.- Borrar caracteres de una oración (2 puntos)

class : StringUtil
method: String removeAllOccurrences(String str, char ch)
type: static

Debes eliminar todas las ocurrencias del carácter ch en el String str. Por ejemplo tu método debe retornar lo siguiente.

> removeAllOccurrences("Esto es una prueba", 's') returns "Eto e una prueba"
> removeAllOccurrences("---0---", '-') returns "0"

Nota: No debe utilizar el método replace de la clase String.

3.- Contar la cantidad de palabras de un texto(2 puntos)

class : StringUtil
method: int countWordsOfAText(String text)
type: static

Debes retornar el número de palabras que tiene un texto.
Ejemplo:

countWords(“Hello world ”) returns 2

4.-Multiplicar String’s(2 puntos)

Dado una cadena de caracteres debes retornar lo siguiente: cada número del 0-9 que aparece en el String debe ser reemplazado por el carácter que le sigue tantas veces sea el valor del número. Por ejemplo el String “a3tx2z ” debe retornar “atttxzz” y “2x” debe retornar “xx” .

class :StringUtil
method: String blow(String str)
type : static

5.- Ocurrencia más larga de un carácter(2 puntos)

Determinar la ocurrencia máxima seguida de un carácter, ejemplo xxyyyz debe retornar 3 ya que la máxima ocurrencia seguida es yyy. Otro ejemplo sería xyzx debe retornar 1.

class :StringUtil
method: int maxRun(String str)
type : static

6.- Crear un árbol binario a partir de una expresión matemática-Opcional(2 puntos).

Tomando en cuenta la siguiente expresión matemática

2+2*4 = 10

el orden de realizar las operaciones sería primero 2*4 y luego el resultado se sumará con el 2.

Tomando otra expresión

(2+2)*4 = 16

el orden de las operaciones matemáticas sería diferente debido al separador, primero se haría la suma 2+2 y luego el resultado se multiplicará con el 4.

Si tomamos en cuenta el orden de operación notariamos que la multiplicación tiene preferencia sobre la suma, a menos de que haya un separador de por medio.

Ahora para a partir de una expresión matemática elaborar un árbol binario tenemos que separar la expresión a partir de los operadores pero la preferencia va ha ser inversa a las de orden de operación de álgebra; es decir, si en álgebra la preferencia se da de esta

forma * y / tienen mayor preferencia que + y - la preferencia que utilizaremos que en este caso llamaremos preferencia de separación será la siguiente:

= + - * / ^

donde = tiene la mayor preferencia y ^ tiene la menor.

Desarrollemos esa lógica con las dos expresiones que hemos visto :

2+2*4

2 2 *4

2 4

ahora con la otra :

(2+2)*4

2+2 4

2 2

Debes considerar los siguientes separadores)] }.

IMPORTANTE

Los programas deben ser enviados al siguiente correo edu.urbe.java@gmail.com antes del 10 de agosto. Prueben con varias entradas para detectar algún error antes de enviar las soluciones, pueden enviar hasta 2 veces la asignación. Debes utilizar el correo que tienes como primario en el sistema de urbe, es decir, el que diste cuando te inscribiste.**TIPS**

En la clase String existen métodos de mucha ayuda para resolver estos problemas, como lo son equals, indexOf, substring y replace. Con el ejercicio del árbol traten de dividir el problema en pequeños pasos(DIVIDE AND CONQUER).

REFERENCIAS

CS 106A Stanford University.

CS 108 Stanford University.