Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Lenguajes Formales y de Programación
Proyecto #2. Segundo semestre 2007

Se tiene un archivo texto con cualquier tipo de texto. Dentro del texto pueden venir declaraciones de variables, operaciones matemáticas y las operaciones de escritura de variables. Estas operaciones pueden estar en cualquier parte del archivo y repetirse cualquier número de veces.

La declaración de variables se realiza mediante la palabra VAR seguido de una lista de variables entre paréntesis. Por ejemplo: VAR (Total, Num, A). Total, Num y A, son las variables definidas. Todas las variables al ser definidas son inicializadas con el valor de 0. Las operaciones matemáticas permitidas son las siguientes:

SUMA: Suma
PROM: Promedio
CONT: Número de elementos en la lista
MAX: Número mayor de la lista
MIN: Número menor de la lista.

Para indicar que viene una operación, debe venir la palabra OPER luego una variable declarada previamente, seguido de una lista de valores constantes o variables entre paréntesis. Todas las operaciones y la declaración de variables van a venir en medio de los siguientes caracteres: {% %}. Por ejemplo: **{%OPER Total SUM (3,4,2,1,A)%}**

Cuyo resultado es 10, producto de sumar la lista de valores. Y la variable Total tendría el valor de 10. Los valores almacenados en las variables pueden ser enteros ó decimales Por ejemplo 10.24, 10 ó 10.0 son validos.

La operación de escritura de variables se utiliza mediante la instrucción ESCRIBIR seguido de una ó más variables ó valores constantes, separados por comas entre paréntesis.
Por ejemplo: **{%ESCRIBIR (Total, A)%}**, debe escribir 10.0.

El proyecto consiste en leer el archivo de texto y generar un nuevo archivo de salida, con el mismo nombre pero con extensión: "*.htm"*, también se debe poder abrir dentro de la aplicación. Este archivo nuevo debe ser generado bajo las siguientes reglas:

1. Debe copiar todo el texto encontrado en el archivo de entrada, tal y como fue leído.
2. La declaración de variables no debe ser copiada al archivo de salida.
3. Cuando se encuentre una operación, esta debe ejecutarse pero no debe ser colocado en el archivo de salida.
4. Cuando se encuentre una instrucción ESCRIBIR, se debe sustituir por el valor de las variables que se encuentran a continuación de la instrucción. Si hay más de una variable, los valores deben ser escritos separados por comas.
5. Si se encuentra un error en el archivo, se debe seguir procesando el resto de archivo ignorando las instrucciones con error. Al final del archivo de salida, se debe escribir la palabra ERRORES: y luego colocar la lista de errores encontrados. Por ejemplo:

ERRORES:

1. <LINEA>,<POSICIÓN>: Variable <nombre de la variable> no fue declarada.
2. <LINEA>,<POSICIÓN>: No existen variables para escribir
3. <LINEA>,<POSICIÓN>: No existe variable para colocar la operación matemática

Se deberá colocar en la documentación la lista de errores posibles, este es solo un ejemplo.

El editor de texto debe resaltar las operaciones que se encuentren dentro de {% %}.

Restricciones:

1. En la documentación se debe incluir las expresiones regulares, y el DFA utilizado.
2. Se debe resolver mediante el método del árbol.
3. Generar la gramática utilizada para el análisis sintáctico para evaluar las operaciones, empleando el autómata de pila.
4. Todo el análisis léxico y sintáctico se debe generar manualmente sin ayuda de una analizador ya definido (Java Lex Y Cup, Flex y Bison, Gold Parser C#Cup, etc).
5. El proyecto debe realizarse en visual Java en cualquier IDE.
6. En ningún momento de la aplicación debe aparecer: *Null Pointer Exception*, de lo contrario se restarán 10 puntos de la calificación obtenida.
7. El proyecto deberá ser entregado con su respectiva documentación, Manual de Usuario y Manual Técnico (en digital nada impreso de lo contrario se restarán 5 puntos), de igual manera deberá contener su respectiva documentación interna (comentarios) para tener derecho a revisión.

Fecha de Entrega: Sábado 3 de noviembre 2007

A continuación se coloca un archivo de entrada y un archivo de salida como ejemplo.

Archivo de entrada: carta.txt

Estimado Señor López
Ciudad de Guatemala
Presente # código: /?(&â™«//

A continuación le envió la lista de notas del estudiante # de carné 10001 para que realice una evaluación de su desempeño **{%VAR (NotaFinal)%}** y me indique que opinión tiene**. {%VAR (MejorNota, Num)%}** En función de sus recomendaciones se procederá a la asignación de la beca al estudiante.

Es importante hacer notar **{%OPER NotaFinal PROM (100,90, 80,95)%}** que el estudiante mantiene un promedio de **{%ESCRIBIR (NotaFinal)%}** y su mejor nota **{%OPER MejorNOta MAX (100,90, 80,95)%}** es de **{%ESCRIBIR (MejorNota)%}**, si compara el promedio con la mejor nota **{%ESCRIBR(NotaFinal,MejorNota)%}**, no hay mucha diferencia. Para estos cálculos se han evaluado **{%OPER Num CONT (100,90, 80,95) ESCRIBIR (Num)%}** notas.

Agradeciendo su comprensión Atentamente **{%ESCRIBIR (Total)%}**

El ejemplo.

Archivo de salida: carta.htm

Estimado Señor López
Ciudad de Guatemala
Presente # código: /?(&â™«//

A continuación le envió la lista de notas del estudiante # de carné 10001 para que realice una evaluación de su desempeño y me indique que opinión tiene. En función de sus recomendaciones se procederá a la asignación de la beca al estudiante.

Es importante hacer notar que el estudiante mantiene un promedio de 91.25 y su mejor nota es de 100, si compara el promedio con la mejor nota 91.25,100, no hay mucha diferencia. Para estos cálculos se han evaluado 4 notas.

Agradeciendo su comprensión Atentamente

El ejemplo.

/**********/
ERRORES:
1. 13, 53: La variable Total no existe