TRABAJO PRACTICO N1

Autómatas

12 DE SEPTIEMBRE DE 2021

[IGNACIO BISIO Y JUAN MOSCATELLI]
Sintaxis y Semántica de los lenguajes - K2055 - UTN FRBA

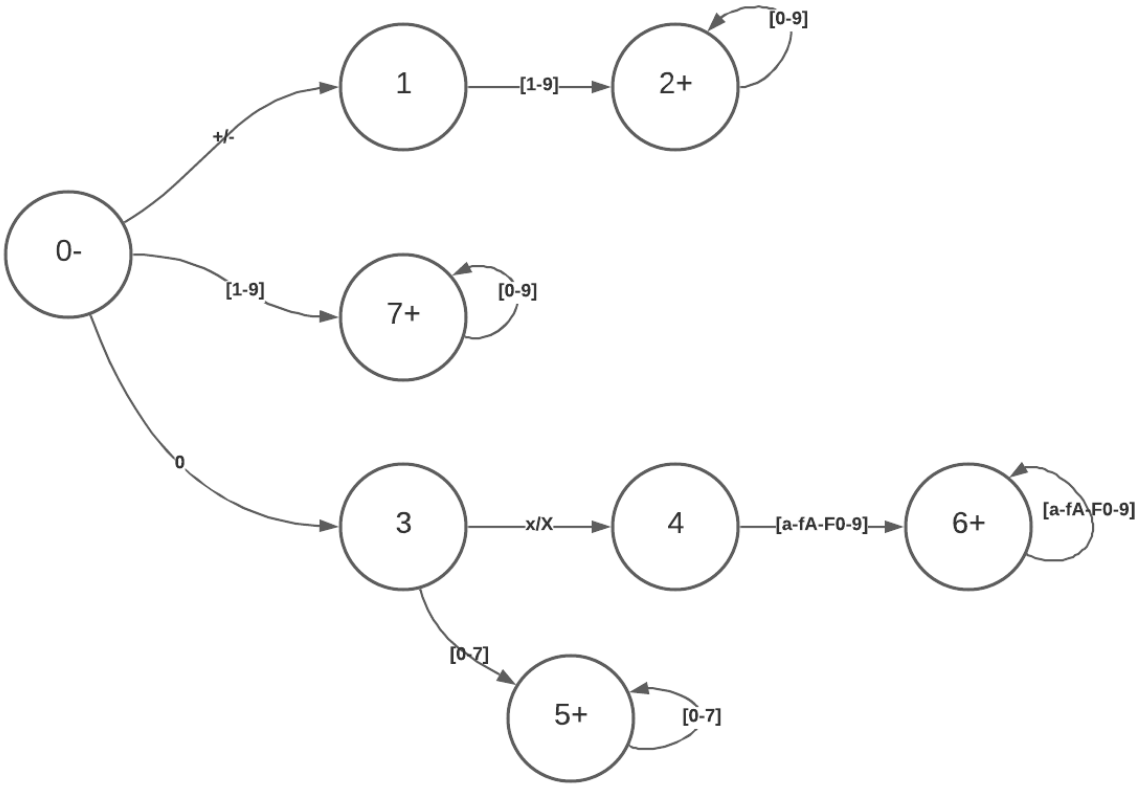
Tabla de contenido

Punto 1

Consigna

Dada una cadena que contenga varios números que pueden ser decimales, octales hexadecimales, con o sin signo para los decimales, separados por el carácter '&', reconocer los tres grupos de constantes enteras, indicando si hubo un error léxico, en caso de ser correcto, contar la cantidad de cada grupo.

- Autómata utilizado: Como el autómata tiene que reconocer 3 tipos de números, lo encaramos de la siguiente forma: Si empieza con un +/- sabemos de qué si o si es un decimal con signo y solo puede tener primero un digito del 1-9 y luego puede tener más dígitos del 0-9.
  Si empieza con un digito del [0-9], significa que es un decimal sin signo, que luego puede tener más dígitos del 0-9
  Si empieza con un 0, significa que puede ser octal o hexadecimal. Por lo que, si a ese 0 le sigue una x o X, significa que es un numero hexadecimal que puede estar seguido los caracteres [a-fA-F0-9]. Pero si al primero 0 en vez de seguirle una x o X, le seguía otro numero del 0-7, significa que es octal.



- Tabla de transición:

Estado	+/-	0	[1-7]	[8-9]	xX	[a-fA-F]	Rechazo
0-	1	3	7	7	8	8	8
1	8	8	2	2	8	8	8
2+	8	2	2	2	8	8	8
3	8	5	5	8	4	8	8
4	8	6	6	6	8	6	8
5+	8	5	5	8	8	5	8
6+	8	6	6	6	8	6	8
7+	8	8	7	7	8	8	8
8	8	8	8	8	8	8	8

- Metodología:

Para este punto vamos a utilizar 2 archivos .txt: “Entrada.txt” (en el que se va a poner la cadena de caracteres que se quiere analizar, ej: 0xAF&111&+55&+8f&0675&+065) y salida .txt (muestra de que tipo es cada palabra, ej: Hexadecimal, Decimal, Decimal, No reconocido, Octal, No Reconocido).

Empezamos leyendo el archivo y mientras el archivo no este vacio y no sea el final de la cadena del archivo, vamos a ir obteniendo carácter a carácter con la función fgetc, y si ese carácter no es un centinela ‘&’, se este en un estado distinto al de rechazo y no sea el final del archivo, nos vamos a mover por la matriz dependiendo del carácter que sea con la función posCol, la cual nos va devolviendo la columna de los caracteres en los siguientes estados y los vamos mostrando con un printf.

Luego de leer toda la palabra, hasta encontrar un centinela o el fin del archivo en caso de ser la última palabra, el contadorDePalabras se va a aumentar 1 en 1 por cada Token leído, mientras este no sea un centinela o el fin del archivo. Y con la función cargarArchvio, nos vamos a encargar de aumentar los contadores de cada tipo de palabra, ya sea Decimal, Octal, Hexadecimal o un No reconocido.

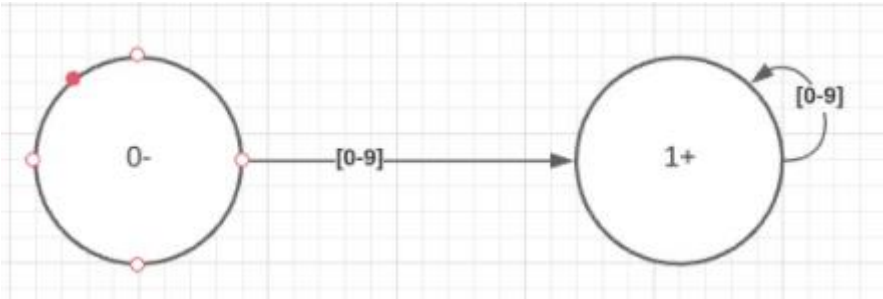
Luego de leer todo el archivo, vamos a generar en el archivo “Salida.txt”, vamos a mostrar de que tipo era cada palabra, ej: Hexadecimal, Decimal, Decimal, etc y vamos a mostrar por consola la cadena total leída, y la cantidad de Tokens con la cantidad de cada tipo.

Punto 2

Consigna

Utilizando el ejercicio 1, ingresar una cadena de caracteres que represente una operación simple con enteros decimales y obtener su resultado, se debe operar con +,- y *. Ejemplo = 3+4*7+3-5=29.

- Autómata utilizado: Este autómata es mucho mas simple que el del punto 1, ya que lo único que queremos que reconozca son enteros decimales.



- Tabla de transición:

Estado	[0-9]	Resto
0-	1	2
1+	1	2
2	2	2

- Metodología:

Al igual que en el punto 1, vamos a trabar con un archivo .txt (calculadora.txt), en donde se va a poder establecer la cuenta que se quiera realizar ej: 4-3*8+5+10.

Creamos una constante llamada numero_actual, la cual va a actuar como acumulador de los dígitos que vayamos leyendo del archivo concatenándose como strings. Para esto, usamos la función strdup que devuelve un string que sea igual a lo que se le pida, pero crea una propia posición de memoria.

Luego, mientras no sea el fin del archivo, obtenemos los caracteres. Mientras esos caracteres no estén en un estado 2 (rechazo) y no sea el final del archivo, mediante la función posCol, nos vamos moviendo por las columnas de la matriz (si es un carácter del [0-9] nos movemos por la columna 0 y si es cualquier otro nos vamos a la columna 1.

Ahora, cuando se obtiene un carácter, se fija mediante la función noEsOperador si era un dígito y lo va concatenando dentro del acumulador numero_actual con la función strncat (dentro de esta función usamos strlen para obtener la longitud del numero y le sumamos 1 para que tenga espacio el \0 de c y no le quite el último digito).

Cuando reconoce una palabra, sale del while y significa que leyó una constante decimal el cual se deriva en 2 caminos: puede ser la primera vez que reconoce o la segunda.

Si es la primera vez, significa que no hay un número guardado antes, pero si es la segunda vez, significa que ya había un número guardado y que hay que operarlo con el numero anterior.

La función que se va a encargar de determinar si fue la primera vez que leímos algo o no, va a ser el if con la variable Flag, la cual fue inicializada en 1, pero luego de leer la primera palabra, se cambia su valor a 0 y se guarda el valor de la primera palabra leída en la variable resultado mediante la función atoi (transofrma un string en int) y en operador vamos a guardar el carácter a utilizar con el siguiente número.

Con la función free, liberamos la posición de memoria del numero_actual para que no consuma mucha más ram y volvemos a establecer que numero_actual este vació para guardar la siguiente palabra.

Esto se vuelve a repetir hasta encontrar el siguiente numero entero y leer otro signo. Ahora, siempre que entra al if(flag) va a realizar la operación mediante la función operar, la cual utiliza la variable global del numero_actual y crea una variable entera (operando) transformando al numero actual en un int y dependiendo de que símbolo de operación sea (+ - *) realiza la operación establecida y lo va guardando en resultado para seguir operando con los siguientes números hasta que sea el final del archivo, y finalizamos mostrando la variable ‘resultado’ con el resultado final.