Tecnológico de Monterrey

“Actividad 3.2 Programando un DFA"

Implementación de métodos computacionales (Gpo 601)

Profesor: Pedro Oscar Pérez Murueta

Jordana Betancourt Menchaca A01707434

Fermín Méndez García- A01703366

19 de marzo del 2022

Jordana Betancourt Menchaca A01707434
Fermín Méndez García- A01703366

Índice

Git Hub →
https://github.com/FerminMendez/Metodos-Computacionales/tree/main/Lexer

1. Documentación.

Acerca del programa.

Este programa es la implementación de un DFA (Autómata finito determinista) que logra identificar algunas expresiones aritméticas en un lenguaje libre de contexto.

Los tokens identificados y sus reglas son las siguientes.

Las expresiones aritméticas sólo podrán contener los siguientes tipos de tokens:

- Enteros: Cadenas de puros dígitos
- Flotantes (Reales): Pueden tener punto decimal o notación científica por ejemplo 0.1, 3.5e-10, 6e2 ó .8
- Operadores: = , + , - , / , *, ó ^
- Identificadores: Variables
- Símbolos especiales: ‘(‘ ó ‘)’
- Comentarios: // seguido de caracteres hasta que se acabe el renglón

Está implementado en el lenguaje C++. Como parte de la documentación se incluyen algunas librerías que nos facilitan las siguientes funciones dentro del programa:

Lectura de archivos de texto y acceso a documentos locales:
#include<fstream>
#include <sstream>

Manejo de expresiones regulares e identificación de tokens válidos.
#include <regex>

El programa gira en torno a esta segunda librería ya que nos permite definir reglas con expresiones regulares que no permiten identificar los tokens dados por las reglas del lenguaje.

1.1 ¿Cómo correr el programa?

Para hacer funcionar el programa es necesario tener un archivo de texto “example.txt.”. Recomendable ponerlo en la misma carpeta que el programa para no tener que buscarlo en otra carpeta.
Posteriormente habrá que abrir una nueva terminal y escribir los siguientes comandos:

g++ main.cpp

Se generará un archivo a.exe

Después se tendrá que emplear el comando ./a.exe nombreArchivo

1.2 Ejemplo para correr el programa

g++ main.cpp

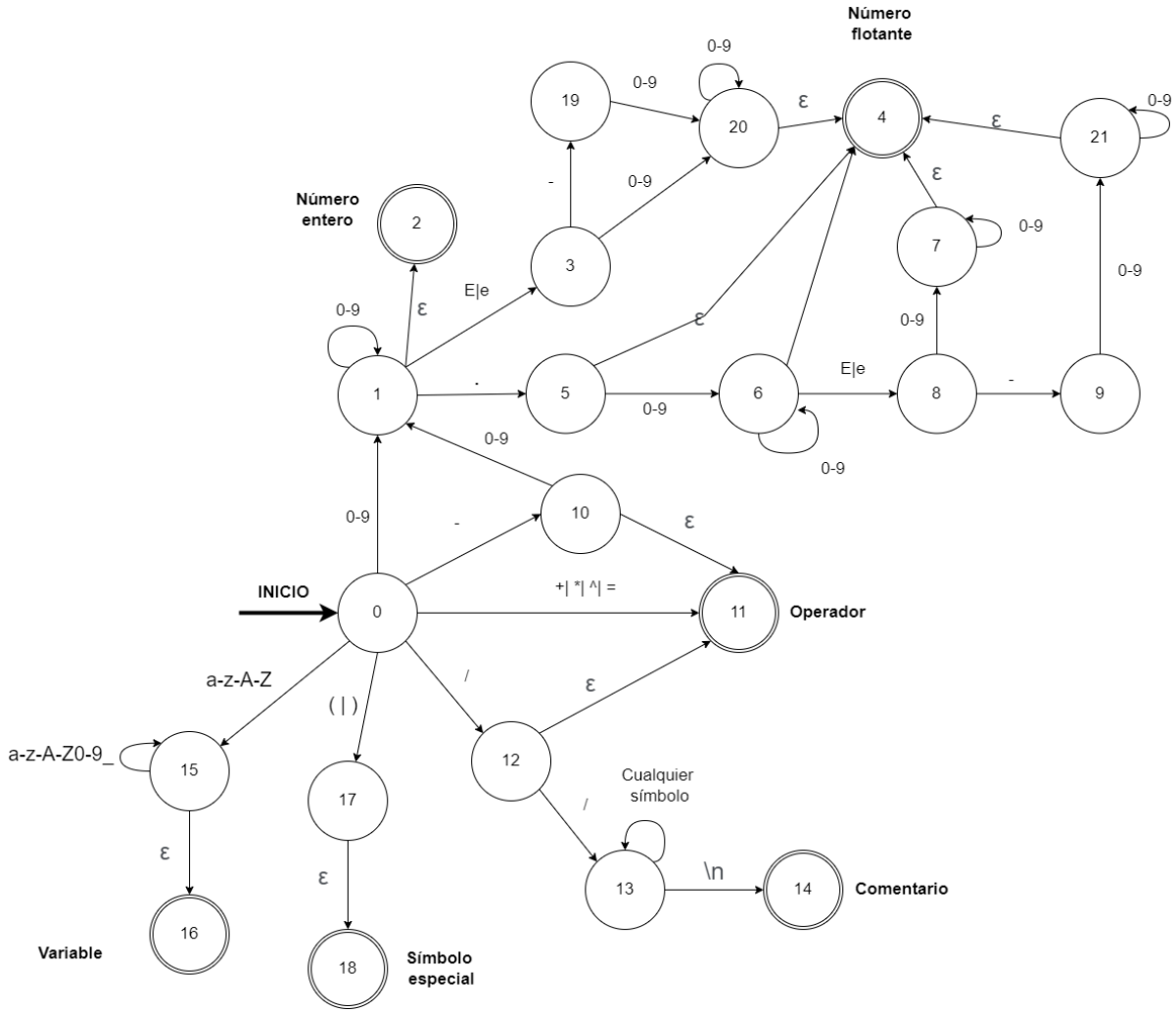
./a.exe input1.txt

1.3 Ejemplo salida del programa

```
b | TOKEN DE VARIABLE
-------------------------------------------
= | TOKEN OPERADOR -> Asignaci├│n
-------------------------------------------
7 | TOKEN DE N├ÚMERO ENTERO
-------------------------------------------
a | TOKEN DE VARIABLE
-------------------------------------------
= | TOKEN OPERADOR -> Asignaci├│n
-------------------------------------------
32.4 | TOKEN DE N├ÚMERO REAL/FLOTANTE
-------------------------------------------
* | TOKEN OPERADOR -> Multiplicaci├│n
-------------------------------------------
( | TOKEN ESPECIAL -> Par├®ntesis izquierdo
-------------------------------------------
-8.6 | TOKEN DE N├ÚMERO REAL/FLOTANTE
-------------------------------------------
- | TOKEN OPERADOR -> Resta
-------------------------------------------
b | TOKEN DE VARIABLE
-------------------------------------------
) | TOKEN ESPECIAL -> Par├®ntesis derecho
-------------------------------------------
```

2. Autómata que resuelve el problema.

El siguiente es el autómata finito no determinista que resuelve el problema:



Jordana Betancourt Menchaca A01707434
Fermín Méndez García- A01703366

3. Tabla de transiciones.

	0-9	E/e	.	-	+	*	/	^	=	a-zA-Z	\n	Cualquier símbolo	(	)
A	B	0	0	**C**	**D**	**D**	**E**	**D**	**D**	**F**	0	0	**G**	**G**
B	**B**	H	**I**	0	0	0	0	0	0	0	0	0	0	0
C	**B**	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	D	0	0	0	0	J	0	0	0	0	J	0	0
F	**F**	0	0	0	0	0	0	0	0	**F**	0	0	0	0
G	0	0	0	0	0	0	0	0	0	0	0	0	0	0
H	**K**	0	0	**L**	0	0	0	0	0	0	0	0	0	0
I	**M**	**0**	0	0	0	0	0	0	0	0	0	0	0	0
J	0	0	0	0	0	0	0	0	0	0	**N**	J	0	0
K	**K**	0	0	0	0	0	0	0	0	0	0	0	0	0
L	**K**	0	0	0	0	0	0	0	0	0	0	0	0	0
M	**M**	O	0	0	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	0
O	**P**	0	0	**Q**	0	0	0	0	0	0	0	0	0	0
P	**P**	0	0	0	0	0	0	0	0	0	0	0	0	0
Q	R	0	0	0	0	0	0	0	0	0	0	0	0	0
R	**R**	0	0	0	0	0	0	0	0	0	0	0	0	0

4. Autómata finito determinista.