# Programming a DFA

## Purpose

One of the Deterministic Finite Automatons applications is the implementation of a token recognizer in a programming language (Also known as a Lexer in compilers).

#### Token Types

* Integer
* Float (Real Numbers)
* Operators
  + Assignation
  + Addition
  + Subtraction
  + Multiplication
  + Division
  + Power
* Identifier
  + Variable
* Special Symbol
  + (
  + )
* Comments

## Information

#### Programming Language

Python

#### Instructions

1. Download Python
2. Download the Zip File
3. Decompress the Zip File
4. You can modify the txt.file

#### Approach

I decided to tackle the DFA with a transition table using dictionaries in python. This way I could easily traverse the states and work directly with my diagram and transition table. By using conditionals, y moved through each state and managed to solve the DFA.

#### Example

#### )8-3.5(

#### a = 32.4 \*(-8.6 - b)/ 6.1E-8

#### d = a ^ b // Esto es un comentario

#### -5987832.3456E-2385 + (Actividades)

#### // Automata Deterministico b = 7

#### )(+\*-^^)

#### final = 36.3461E890/.049\*(10)

#### asdjkl + 10.

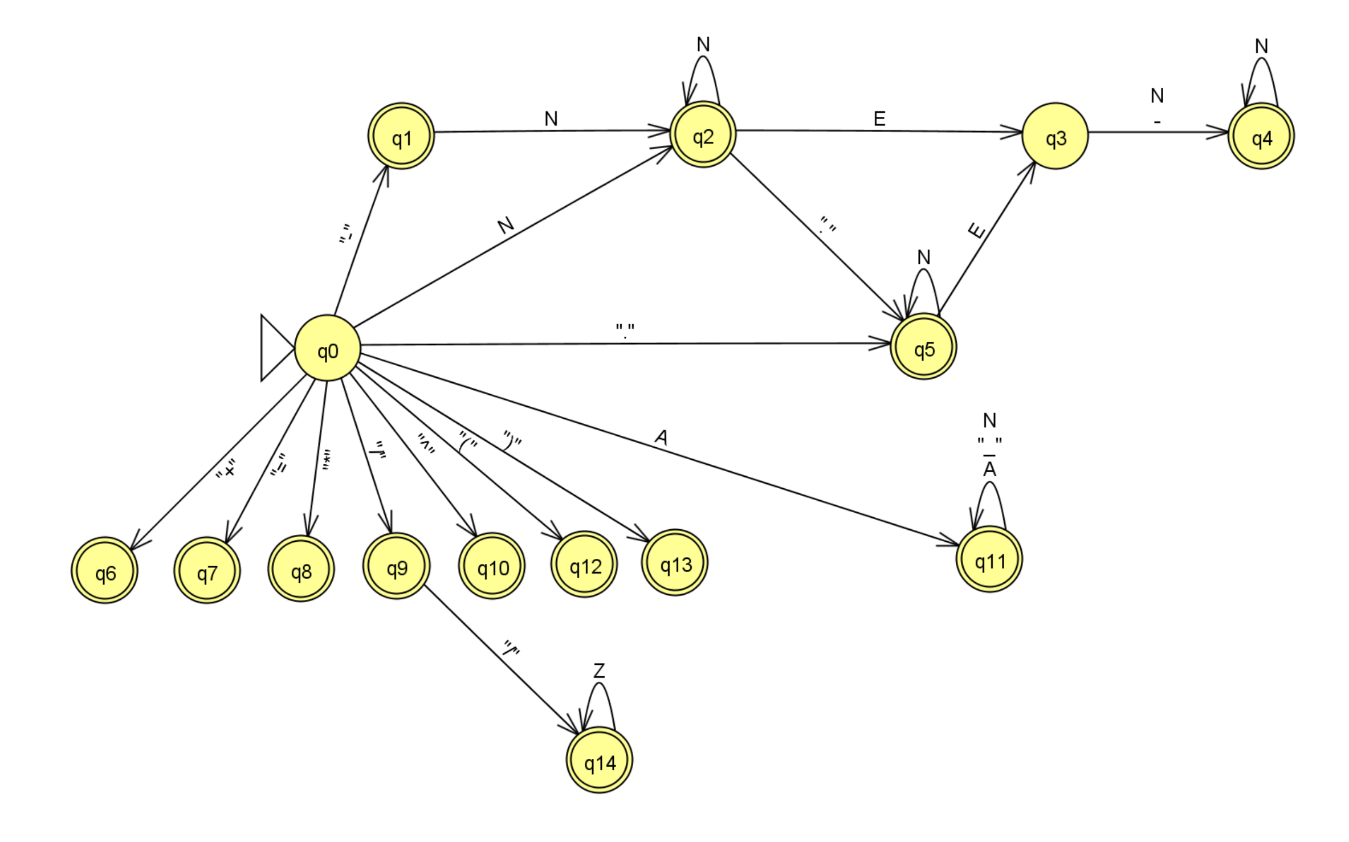
#### Results

Texto

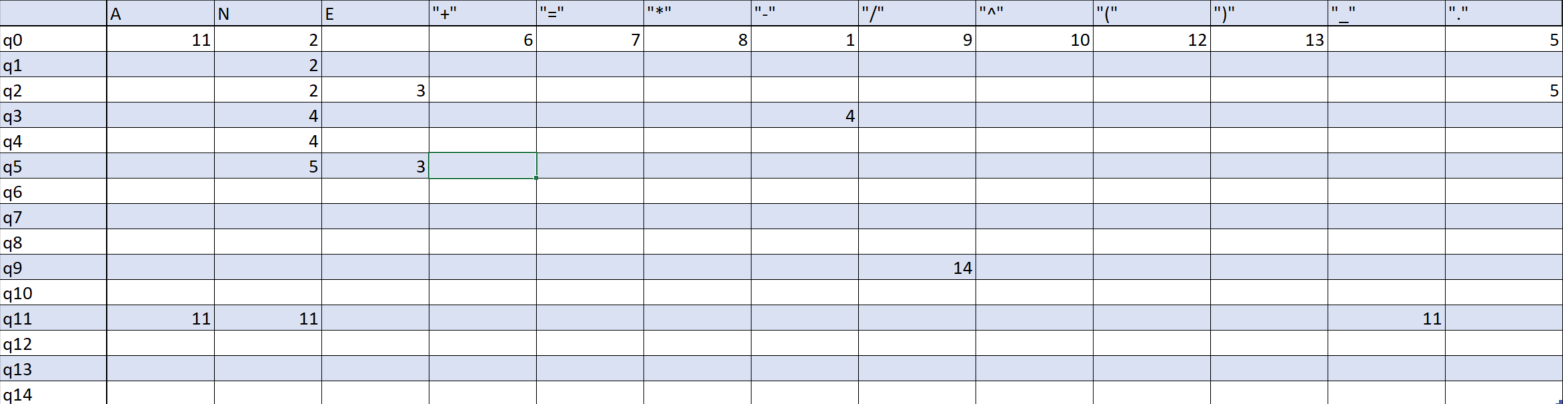
Descripción generada automáticamente

## Texto Descripción generada automáticamente con confianza baja

## Deterministic Finite Automaton

[](https://user-images.githubusercontent.com/71723297/158081102-c3ade6d0-e600-4f79-8933-1a24a075c6a7.png)

## Transition Table

[](https://user-images.githubusercontent.com/71723297/158081378-16440fa3-1506-4279-aaaa-da2cd629f483.png)