

CAUSWE Compiler, 2021

# Compiler Project #1

## Documentation

---



**Class#01 Team#59 안재형(20171248)**

**Class#02 Team#56 구건모(20181856)**

---

---

## Definition of Tokens

Signed integer

<SIGNED INTEGER> - for non-empty sequence of digit, starting from a non-zero digit or minus sign symbol. And single zero digit e.g 0, 11, -123

Single character

<CHARACTER> - for a single digit, English letter, block or any symbol starting from and terminating with ' e.g '0', 'a'

Boolean string

<BOOLSTR> - for true and false

Literal string

<LITERAL STRING> - for any combination of digits, letters, blank, starting from and terminating with " e.g "Hello"

An identifier of variables and functions

<ID> - for a non-empty sequence of English letters, digits, and underscore starting from an English letter or a underscore e.g i, j, \_func\_.

Keywords for special statements

<KEYWORD> - for 'if', 'else', 'while', 'class', 'return' keyword

Arithmetic operators

<ARITHMETICOP> - for '+', '-', '\*', '/', operator

Assignment operator

<OP> - for assign operator '='

Comparison operator

<COMPARISON> - for compare operator '>', '>=', '<', '<=', '==', '!='

A terminating symbol of statements,

A pair of symbols for defining area/scope of variables and functions,

for indicating a function/statement, for using an array,

for separating input arguments in functions.

<SYMBOL> - for ';', '{', '}', '(', ')', '[', ']', ',', '.'

---

## Regular Expression

Regular expression

$\langle \text{VARTYPE} \rangle = \text{int} | \text{char} | \text{boolean} | \text{String}$

$\text{DIGIT} = 0 | 1 | 2 | \dots | 8 | 9$      $\text{Non-zero} = 1 | 2 | 3 | \dots | 9$

$\text{LETTER} = a | b | \dots | y | z | A | B | \dots | Y | Z$

$\text{OTHER}$ : any other symbol ex) '\$', ' ', 'w'

$\langle \text{SIGNED INTEGER} \rangle = ( (- | \epsilon ) \text{Non-zero} (\text{Non-zero} | 0)^* ) | 0$

$\langle \text{CHARACTER} \rangle = \text{'DIGIT | LETTER | OTHER'}$

$\langle \text{BOOLSTR} \rangle = \text{true} | \text{false}$

$\langle \text{LITERAL STRING} \rangle = \text{"(DIGIT | LETTER)*"}$

$\langle \text{ID} \rangle = (\text{LETTER} | \_)(\text{LETTER} | \text{DIGIT} | \_)^*$

$\langle \text{KEYWORD} \rangle = \text{if} | \text{else} | \text{while} | \text{class} | \text{return}$

$\langle \text{ARITHMETIC OP} \rangle = + | - | * | /$

$\langle \text{OP} \rangle = =$

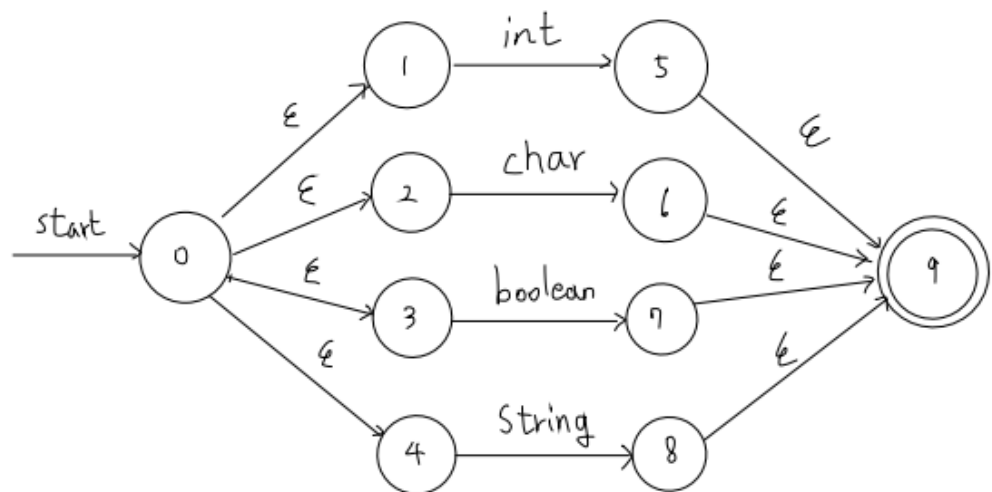
$\langle \text{COMPARISON} \rangle = (( > | < ) ( = | \epsilon ) ) | ( ( ! | = ) = )$

$\langle \text{SYMBOL} \rangle = ; | \{ | \} | ( | ) | [ | ] | ,$

## NFAs

- **<VARTYPE>**

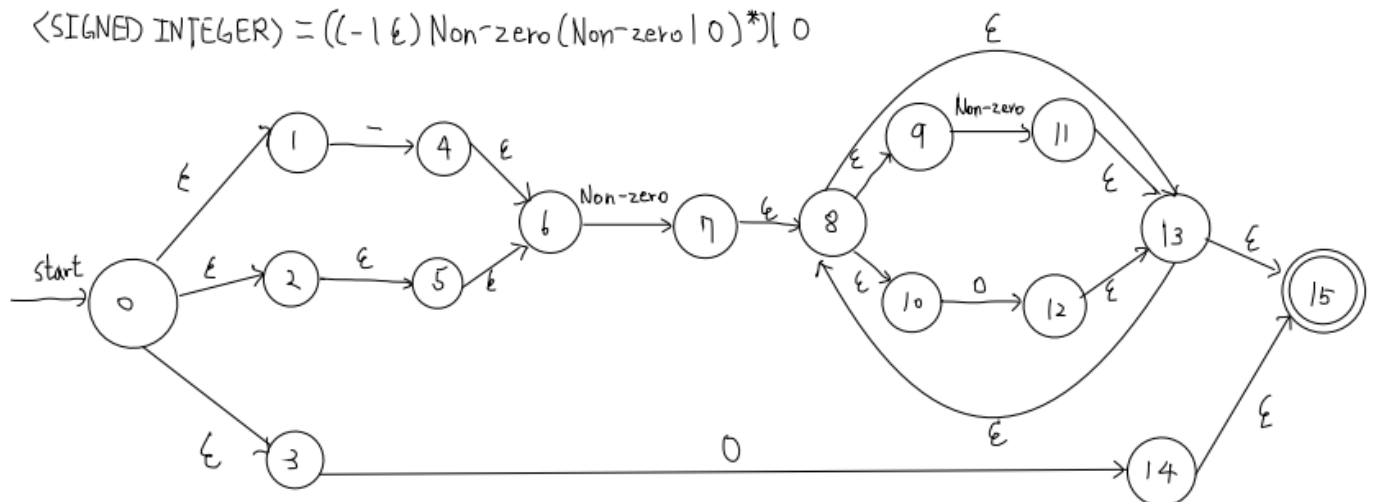
$\langle \text{VARTYPE} \rangle = \text{int} \mid \text{char} \mid \text{boolean} \mid \text{String}$



- **<SIGNED INTEGER>**

$\text{Non-zero} = 1 \mid 2 \mid 3 \mid 4 \mid \dots \mid 8 \mid 9$

$\langle \text{SIGNED INTEGER} \rangle = ((-1 \epsilon) \text{Non-zero} (\text{Non-zero} \mid 0)^*) \mid 0$



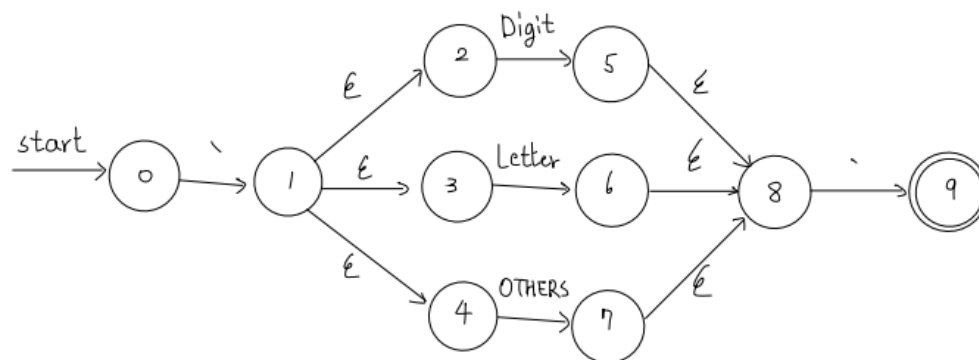
- **<CHARACTER>**

Digit = 0|1|...|8|9

OTHERS: any symbol (e.g. ' , ' \$ , ' % ')

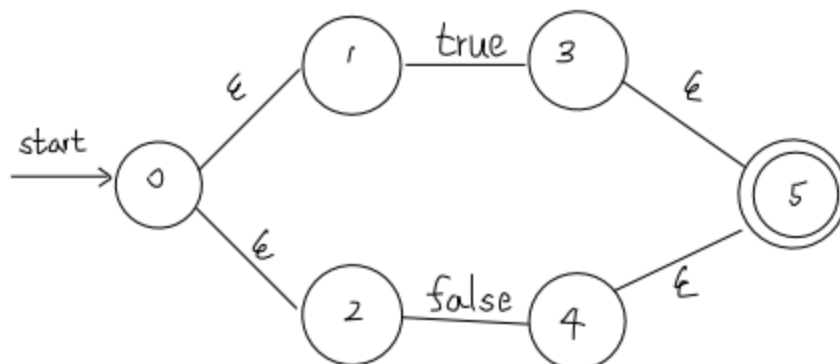
Letter = a|b|...|z|A|B|...|Y|Z

<CHARACTER> = '(Digit|Letter|OTHERS)'



- **<BOOLSTR>**

<BOOLSTR> = true|false

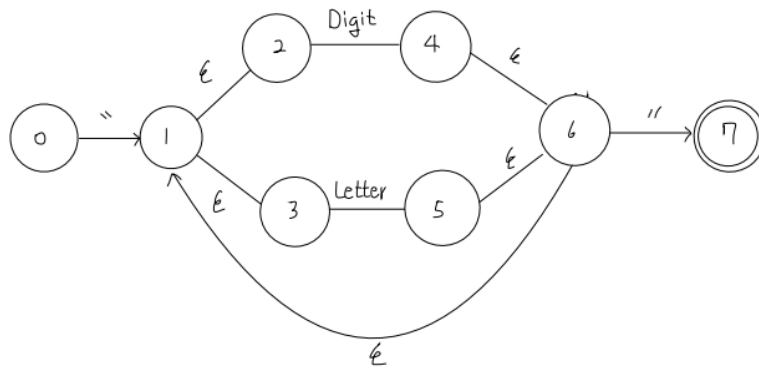


- **<LITERAL STRING>**

Digit = 0|1|2|...|8|9

Letter = a|b|...|z|A|B|...|Z

<LITERALSTRING> = "(Digit|Letter)+"

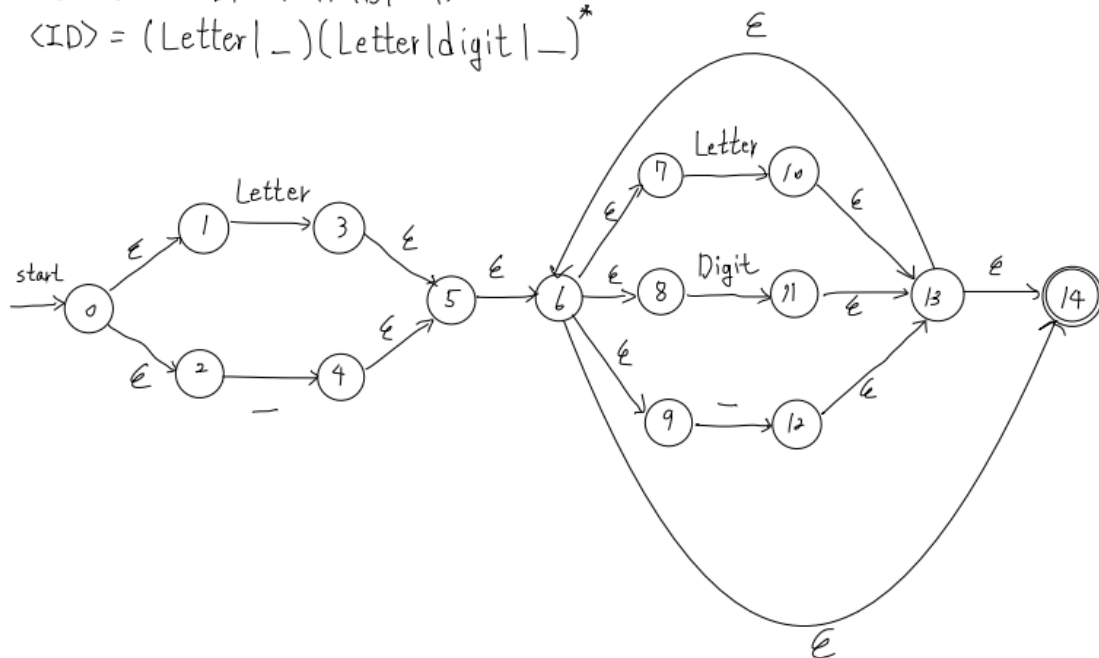


- **<ID>**

Digit = 0|1|...|9

Letter = a|b|...|z|A|B|...|Z

<ID> = (Letter|\_)(Letter|digit|\_)\*

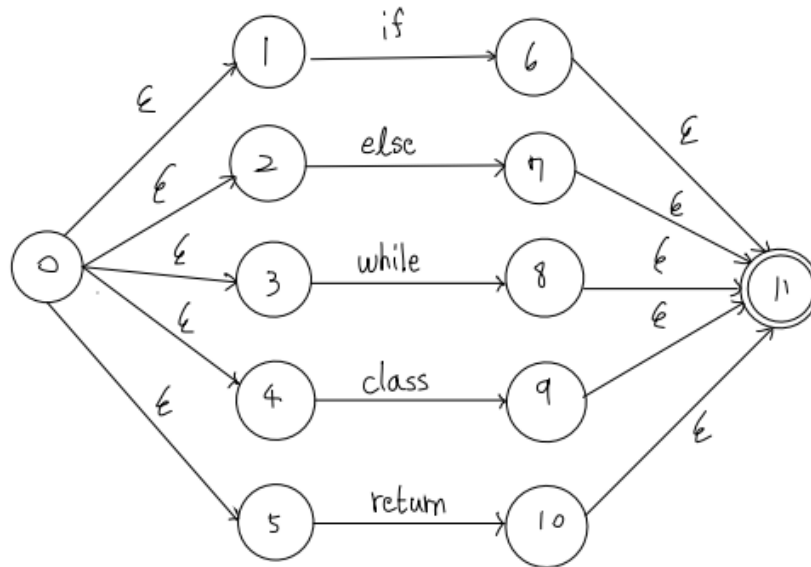


---

- **<KEYWORD>**

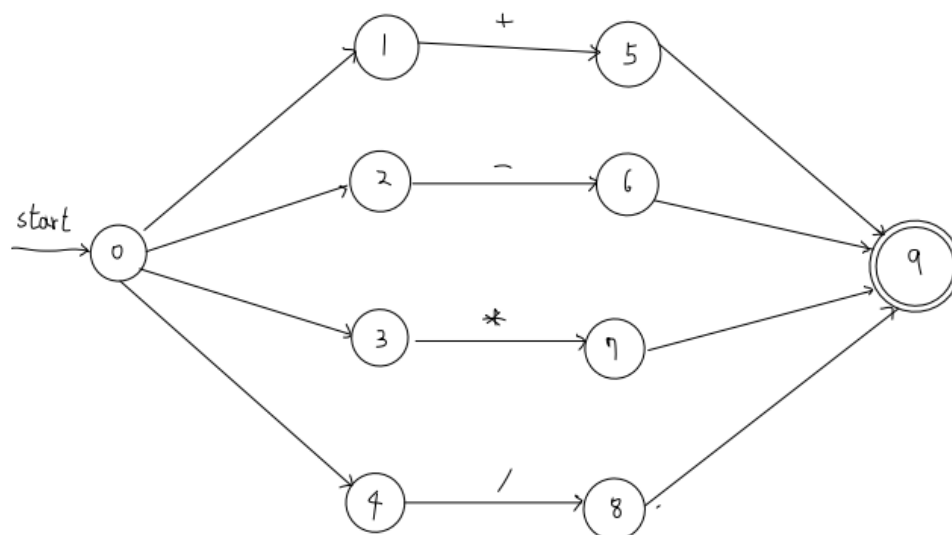
<KEYWORD> NFA

<KEYWORD> = if|else|while|class|return



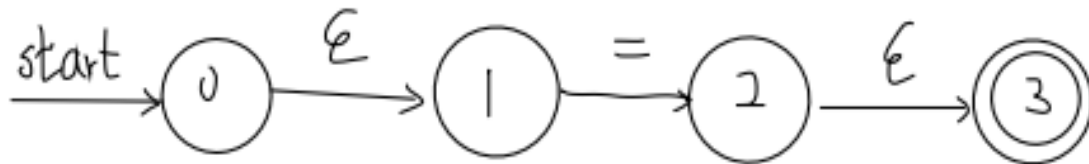
- **<ARITHMETIC OP>**

<ARITHMETIC OP> = +|-|\*|/



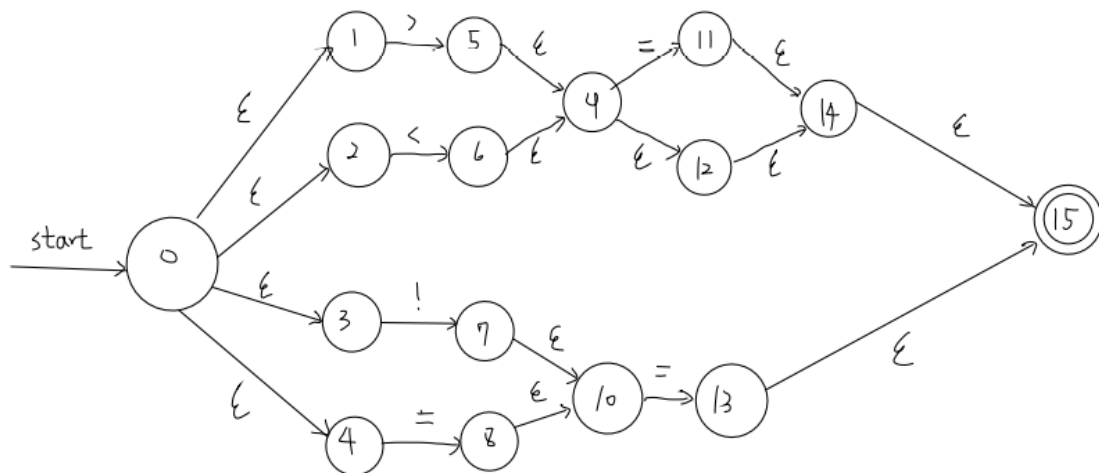
- **<OP>**

$\langle OP \rangle = =$



- **<COMPARISON>**

$\langle COMPARISON \rangle = ((>|<)(=|ε)) | ((!| |=))$

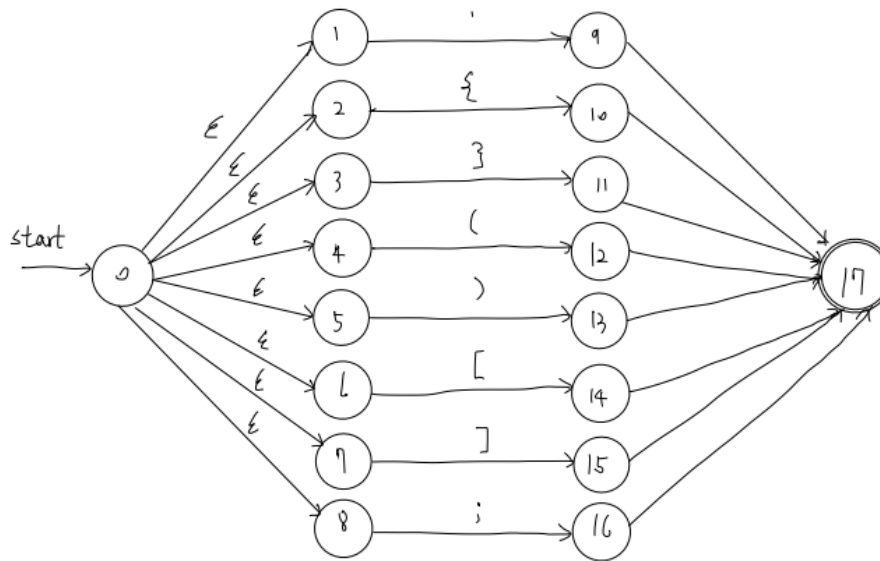




---

- <SYMBOL>

<SYMBOL> = , | { | } | ( | ) | [ | ] | ;



## DFAs and Transition table

### • <VARTYPE>

$$T_0 = \epsilon\text{-closure}(0) = \{0, 1, 2, 3, 4\}$$

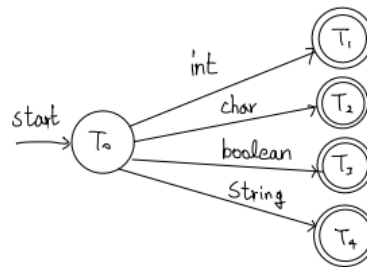
$$T_1 = \epsilon\text{-closure}(\delta(T_0, \text{int})) = \epsilon\text{-closure}(1) = \{1, 5, 9\}$$

$$T_2 = \epsilon\text{-closure}(\delta(T_0, \text{char})) = \epsilon\text{-closure}(2) = \{2, 6, 9\}$$

$$T_3 = \epsilon\text{-closure}(\delta(T_0, \text{boolean})) = \epsilon\text{-closure}(3) = \{3, 7, 9\}$$

$$T_4 = \epsilon\text{-closure}(\delta(T_0, \text{String})) = \epsilon\text{-closure}(4) = \{4, 8, 9\}$$

	int	char	boolean	String
$T_0$	$T_1$	$T_2$	$T_3$	$T_4$
$T_1$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_2$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_3$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$



### • <SIGNED INTEGER>

$$T_0 = \epsilon\text{-closure}(0) = \{0, 1, 2, 3, 5, 6\}$$

$$T_1 = \epsilon\text{-closure}(\delta(T_0, -)) = \epsilon\text{-closure}(4) = \{4, 6\}$$

$$T_2 = \epsilon\text{-closure}(\delta(T_0, \text{Non-zero})) = \epsilon\text{-closure}(7) = \{7, 8, 9, 10, 13, 15\}$$

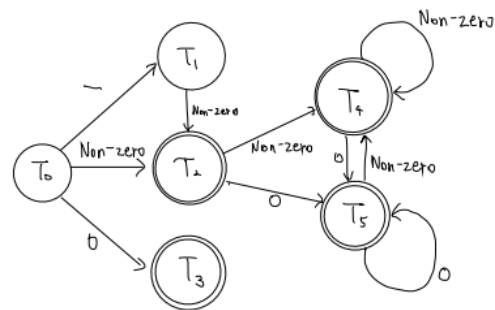
$$T_3 = \epsilon\text{-closure}(\delta(T_0, 0)) = \epsilon\text{-closure}(14) = \{14, 15\}$$

$$\epsilon\text{-closure}(\delta(T_1, \text{Non-zero})) = T_2$$

$$T_4 = \epsilon\text{-closure}(\delta(T_2, \text{Non-zero})) = \epsilon\text{-closure}(11) = \{8, 9, 10, 11, 13, 15\}$$

$$T_5 = \epsilon\text{-closure}(\delta(T_2, 0)) = \epsilon\text{-closure}(12) = \{8, 9, 10, 12, 13, 15\}$$

-	Non-zero	0
$T_0$	$T_1$	$T_3$
$T_1$	$\emptyset$	$\emptyset$
$T_2$	$\emptyset$	$\emptyset$
$T_3$	$\emptyset$	$\emptyset$
$T_4$	$\emptyset$	$\emptyset$
$T_5$	$\emptyset$	$\emptyset$



- **<CHARACTER>**

$$T_0 = \epsilon\text{-closure}(0) = \{0\}$$

$$T_1 = \epsilon\text{-closure}(\delta(T_0, \cdot)) = \epsilon\text{-closure}(1) = \{1, 2, 3, 4\}$$

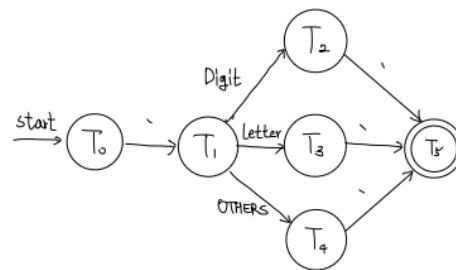
$$T_2 = \epsilon\text{-closure}(\delta(T_1, \text{Digit})) = \epsilon\text{-closure}(5) = \{5, 8\}$$

$$T_3 = \epsilon\text{-closure}(\delta(T_1, \text{Letter})) = \epsilon\text{-closure}(6) = \{6, 8\}$$

$$T_4 = \epsilon\text{-closure}(\delta(T_1, \text{OTHERS})) = \epsilon\text{-closure}(7) = \{7, 8\}$$

$$T_5 = \epsilon\text{-closure}(\delta(T_2, \cdot)) = \epsilon\text{-closure}(8) = \{8, 9\}$$

	$\cdot$	Digit	Letter	OTHERS
$T_0$	$T_1$	$\emptyset$	$\emptyset$	$\emptyset$
$T_1$	$\emptyset$	$T_2$	$T_3$	$T_4$
$T_2$	$T_5$	$\emptyset$	$\emptyset$	$\emptyset$
$T_3$	$T_5$	$\emptyset$	$\emptyset$	$\emptyset$
$T_4$	$T_5$	$\emptyset$	$\emptyset$	$\emptyset$
$T_5$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$



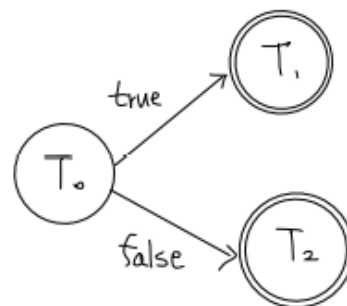
- **<BOOLSTR>**

$$T_0 = \epsilon\text{-closure}(0) = \{0, 1, 2\}$$

$$T_1 = \epsilon\text{-closure}(\delta(T_0, \text{true})) = \epsilon\text{-closure}(3) = \{3, 5\}$$

$$T_2 = \epsilon\text{-closure}(\delta(T_0, \text{false})) = \epsilon\text{-closure}(4) = \{4, 5\}$$

	true	false
$T_0$	$T_1$	$T_2$
$T_1$	$\emptyset$	$\emptyset$
$T_2$	$\emptyset$	$\emptyset$



## • <LITERAL STRING>

$$T_0 = \epsilon\text{-closure}(0) = \{0\}$$

$$T_1 = \epsilon\text{-closure}(\delta(T_0, \epsilon)) = \epsilon\text{-closure}(1) = \{1, 2, 3\}$$

$$T_2 = \epsilon\text{-closure}(\delta(T_1, \text{Digit})) = \epsilon\text{-closure}(4) = \{1, 2, 3, 4, 6\}$$

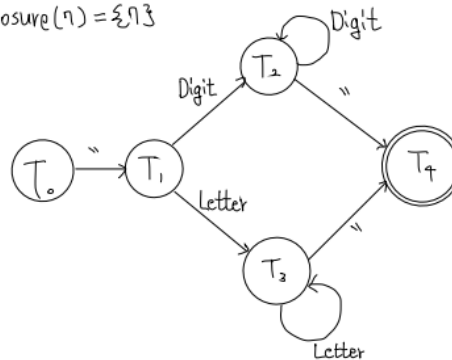
$$T_3 = \epsilon\text{-closure}(\delta(T_1, \text{Letter})) = \epsilon\text{-closure}(5) = \{1, 2, 3, 5, 6\}$$

$$\epsilon\text{-closure}(\delta(T_2, \text{Digit})) = T_2 \quad \epsilon\text{-closure}(\delta(T_2, \text{Letter})) = T_3$$

$$\epsilon\text{-closure}(\delta(T_3, \text{Digit})) = T_2 \quad \epsilon\text{-closure}(\delta(T_3, \text{Letter})) = T_3$$

$$T_4 = \epsilon\text{-closure}(\delta(T_2, \epsilon)) = \epsilon\text{-closure}(7) = \{7\}$$

	$\epsilon$	Digit	Letter
$T_0$	$T_1$	$\emptyset$	$\emptyset$
$T_1$	$\emptyset$	$T_2$	$T_3$
$T_2$	$T_4$	$T_2$	$T_3$
$T_3$	$T_4$	$T_2$	$T_3$
$T_4$	$\emptyset$	$\emptyset$	$\emptyset$



## • <ID>

<ID> NFA to DFA

$$T_0 = \epsilon\text{-closure}(0) = \{0, 1, 2\}$$

$$T_1 = \epsilon\text{-closure}(\delta(T_0, \text{Letter})) = \epsilon\text{-closure}(3) = \{3, 5, 6, 7, 8, 9, 14\}$$

$$T_2 = \epsilon\text{-closure}(\delta(T_0, \epsilon)) = \epsilon\text{-closure}(4) = \{4, 5, 6, 7, 8, 9, 14\}$$

$$T_3 = \epsilon\text{-closure}(\delta(T_1, \text{Letter})) = \epsilon\text{-closure}(10) = \{6, 7, 8, 9, 10, 13, 14\}$$

$$T_4 = \epsilon\text{-closure}(\delta(T_1, \text{Digit})) = \epsilon\text{-closure}(11) = \{6, 7, 8, 9, 11, 13, 14\}$$

$$T_5 = \epsilon\text{-closure}(\delta(T_2, \epsilon)) = \epsilon\text{-closure}(12) = \{6, 7, 8, 9, 12, 13, 14\}$$

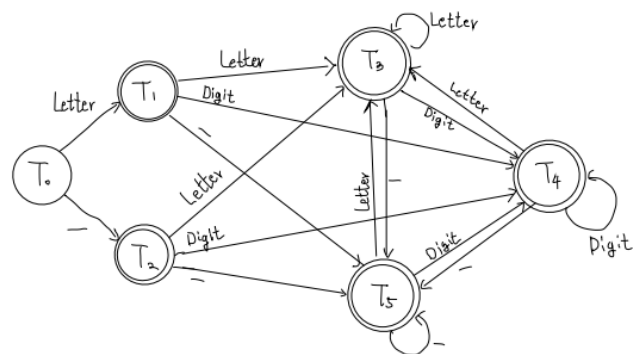
$$\epsilon\text{-closure}(\delta(T_2, \text{Letter})) = T_3 \quad \epsilon\text{-closure}(\delta(T_2, \text{Digit})) = T_4 \quad \epsilon\text{-closure}(\delta(T_2, \epsilon)) = T_5$$

$$\epsilon\text{-closure}(\delta(T_3, \text{Letter})) = T_3 \quad \epsilon\text{-closure}(\delta(T_3, \text{Digit})) = T_4 \quad \epsilon\text{-closure}(\delta(T_3, \epsilon)) = T_5$$

$$\epsilon\text{-closure}(\delta(T_4, \text{Letter})) = T_3 \quad \epsilon\text{-closure}(\delta(T_4, \text{Digit})) = T_4 \quad \epsilon\text{-closure}(\delta(T_4, \epsilon)) = T_5$$

$$\epsilon\text{-closure}(\delta(T_5, \text{Letter})) = T_3 \quad \epsilon\text{-closure}(\delta(T_5, \text{Digit})) = T_4 \quad \epsilon\text{-closure}(\delta(T_5, \epsilon)) = T_5$$

	Letter	Digit	$\epsilon$
$T_0$	$T_1$	$\emptyset$	$T_2$
$T_1$	$T_3$	$T_4$	$T_5$
$T_2$	$T_3$	$T_4$	$T_5$
$T_3$	$T_3$	$T_4$	$T_5$
$T_4$	$T_3$	$T_4$	$T_5$
$T_5$	$T_3$	$T_4$	$T_5$



- **<KEYWORD>**

$$T_0 = \mathcal{E}\text{-closure}(0) = \{0, 1, 2, 3, 4, 5\}$$

$$T_1 = \mathcal{E}\text{-closure}(\delta(T_0, \text{if})) = \mathcal{E}\text{-closure}(6) = \{6, 11\}$$

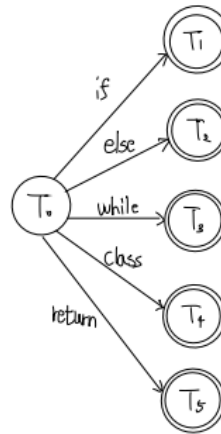
$$T_2 = \mathcal{E}\text{-closure}(\delta(T_0, \text{else})) = \mathcal{E}\text{-closure}(7) = \{7, 11\}$$

$$T_3 = \mathcal{E}\text{-closure}(\delta(T_0, \text{while})) = \mathcal{E}\text{-closure}(8) = \{8, 11\}$$

$$T_4 = \mathcal{E}\text{-closure}(\delta(T_0, \text{class})) = \mathcal{E}\text{-closure}(9) = \{9, 11\}$$

$$T_5 = \mathcal{E}\text{-closure}(\delta(T_0, \text{return})) = \mathcal{E}\text{-closure}(10) = \{10, 11\}$$

	if	else	while	class	return
$T_0$	$T_1$	$T_2$	$T_3$	$T_4$	$T_5$
$T_1$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_2$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_3$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_4$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_5$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$



- **<ARITHMETIC OP>**

$$T_0 = \mathcal{E}\text{-closure}(0) = \{0, 1, 2, 3, 4\}$$

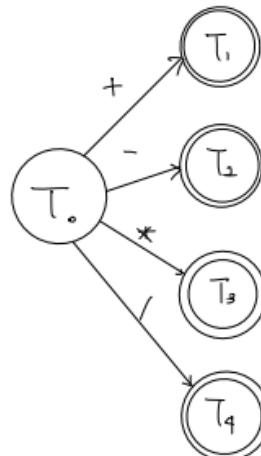
$$T_1 = \mathcal{E}\text{-closure}(\delta(T_0, +)) = \mathcal{E}\text{-closure}(1) = \{5, 9\}$$

$$T_2 = \mathcal{E}\text{-closure}(\delta(T_0, -)) = \mathcal{E}\text{-closure}(2) = \{6, 9\}$$

$$T_3 = \mathcal{E}\text{-closure}(\delta(T_0, *)) = \mathcal{E}\text{-closure}(3) = \{7, 9\}$$

$$T_4 = \mathcal{E}\text{-closure}(\delta(T_0, /)) = \mathcal{E}\text{-closure}(4) = \{8, 9\}$$

	+	-	*	/
$T_0$	$T_1$	$T_2$	$T_3$	$T_4$
$T_1$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_2$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_3$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_4$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

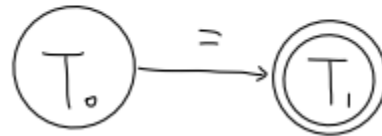


- <OP>

$$T_0 = \epsilon\text{-closure}(0) = \{0, 1\}$$

$$T_1 = \epsilon\text{-closure}(\delta(T_0, >)) = \epsilon\text{-closure}(2) = \{2, 3\}$$

$$\begin{array}{cc} & = \\ T_0 & T_1 \\ T_1 & \emptyset \end{array}$$



- <COMPARISON>

<COMPARISON> NFA to DFA

$$T_0 = \epsilon\text{-closure}(0) = \{0, 1, 2, 3\}$$

$$T_1 = \epsilon\text{-closure}(\delta(T_0, >)) = \epsilon\text{-closure}(5) = \{5, 9, 12, 14, 15\}$$

$$T_2 = \epsilon\text{-closure}(\delta(T_0, <)) = \epsilon\text{-closure}(6) = \{6, 9, 12, 14, 15\}$$

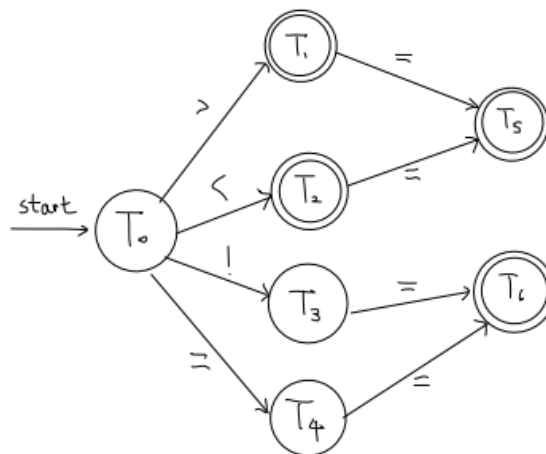
$$T_3 = \epsilon\text{-closure}(\delta(T_0, !)) = \epsilon\text{-closure}(7) = \{7, 10\}$$

$$T_4 = \epsilon\text{-closure}(\delta(T_0, =)) = \epsilon\text{-closure}(8) = \{8, 10\}$$

$$T_5 = \epsilon\text{-closure}(\delta(T_1, =)) = \epsilon\text{-closure}(11) = \{11, 14, 15\}$$

$$T_6 = \epsilon\text{-closure}(\delta(T_3, =)) = \epsilon\text{-closure}(13) = \{13, 15\}$$

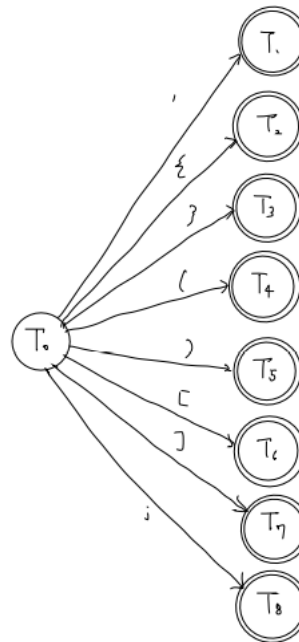
	>	<	!	=
$T_0$	$T_1$	$T_2$	$T_3$	$T_4$
$T_1$	$\emptyset$	$\emptyset$	$\emptyset$	$T_5$
$T_2$	$\emptyset$	$\emptyset$	$\emptyset$	$T_5$
$T_3$	$\emptyset$	$\emptyset$	$\emptyset$	$T_6$
$T_4$	$\emptyset$	$\emptyset$	$\emptyset$	$T_6$
$T_5$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_6$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$



## • <SYMBOL>

$T_0 = \epsilon\text{-closure}(0) = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$   
 $T_1 = \epsilon\text{-closure}(\delta(T_0, ')) = \epsilon\text{-closure}(9) = \{9, 17\}$   
 $T_2 = \epsilon\text{-closure}(\delta(T_0, \epsilon)) = \epsilon\text{-closure}(10) = \{10, 17\}$   
 $T_3 = \epsilon\text{-closure}(\delta(T_0, \{)) = \epsilon\text{-closure}(11) = \{11, 17\}$   
 $T_4 = \epsilon\text{-closure}(\delta(T_0, ( )) = \epsilon\text{-closure}(12) = \{12, 17\}$   
 $T_5 = \epsilon\text{-closure}(\delta(T_0, )) = \epsilon\text{-closure}(13) = \{13, 17\}$   
 $T_6 = \epsilon\text{-closure}(\delta(T_0, [ )) = \epsilon\text{-closure}(14) = \{14, 17\}$   
 $T_7 = \epsilon\text{-closure}(\delta(T_0, ]) = \epsilon\text{-closure}(15) = \{15, 17\}$   
 $T_8 = \epsilon\text{-closure}(\delta(T_0, ;)) = \epsilon\text{-closure}(16) = \{16, 17\}$

	'	$\epsilon$	{	(	)	[	]	;
$T_0$	$T_1$	$T_2$	$T_3$	$T_4$	$T_5$	$T_6$	$T_7$	$T_8$
$T_1$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_2$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_3$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_4$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_5$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_6$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_7$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$T_8$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$



## How This Lexical Analyzer Work?

### Requirements

- Python >= 3.8.2
- Ubuntu or MacOS

### Architecture

There's a `DFA_Graph` class. This class contains a token's DFA Informations and `current_state`. If it gets a symbol as an input, `current_state` changes and returns its value. Of course, If it got a wrong symbol, `current_state` changes to `None` and returns it until it resets.

There are syntaces as `DFA_Graph` class in this code.

---

For the main function,

1. it reads a file by a single symbol.
2. Then pass it into every token DFA\_Graph's input.
3. Repeat it until every DFA\_Graph returns None.
4. Print token's information that returned a number until last time.  
If there were several tokens, choose the first one.
5. From the last symbol, do 2 and repeat until a file ends.

And If parsing ends up with an error, it reports where the error occurred.  
(Error Message is not written at output file.)

## Running Table

This table shows what happens for each symbol input.

		i	n	t		t	e	s	t	
DFA_Graph	INT .current_state	1	1	2(f)	None →None	None →None	None	None	None	None →None
	ID .current_state	2(f)	2(f)	2(f)	None →None	None →1(f)	1(f)	1(f)	1(f)	None →None
	WS .current_state	None	None	None	None →1(f)	None →None	None	None	None	None →1(f)
		Every states becomes None↑ print INT(priority)								
					↑ print WS					
									↑ print ID	



## Sample Result

input (sample2.java)	output (sample2.java_output.txt)
<pre>int func(int a) { return 0; }  </pre>	<pre>INT int ID func LPAREN ( INT int ID a RPAREN ) LBRACE { RETURN return SIGNED INTEGER 0 TERMINATE ; RBRACE }</pre>

input (sample5.java)	output (sample5.java_output.txt)
<pre>if(true) {     "Hello World" }else{     "It's Java Time!" }  </pre>	<pre>IF if LPAREN ( BOOLEAN STRING true RPAREN ) LBRACE { STRING "Hello World" RBRACE } ELSE else LBRACE { STRING "It's Java Time!" RBRACE }</pre>

input (sample_final.java)	output (sample_final.java_output.txt)
<pre>int char boolean String 0 -5 2345 'a' '1' '%' '&amp;' ; true false</pre>	<pre>INT int CHAR char BOOLEAN boolean STRING String SIGNED INTEGER 0 SIGNED INTEGER -5 SIGNED INTEGER 2345 SINGLE CHARACTER 'a' SINGLE CHARACTER '1'</pre>

```

"Hello Java World"
class Foo
a+b
b-c
1-5
-17
int foo = -3;
int _bar_123 = 4;
if(true) {
"Hello World";
}else{
"It's Java Time!";
}
int func(int a) {return a}
int[] a = {1,2,3};
class compiler {
    String Teacher;
    char Grade;
}
3 > 5
3 >= 5
3 < 5
3 <= 5
3 != 5
3 == 5
int b = 3 * 5
b = 3 / 5
int i = 4;
while(i < 5){
    i = 10;
}
"dddddsdfasdfasdf"
ghfhhdh"
This Line is made for Error

```

```

SINGLE CHARACTER      '%'
SINGLE CHARACTER      '&'
TERMINATE            ;
BOOLEAN STRING true
BOOLEAN STRING false
STRING  "Hello Java World"
CLASS    class
ID       Foo
ID       a
OP_ARITHMETIC  +
ID       b
ID       b
OP_ARITHMETIC  -
ID       c
SIGNED INTEGER 1
OP_ARITHMETIC  -
SIGNED INTEGER 5
SIGNED INTEGER -17
INT         int
ID         foo
OP_ASSIGNMENT =
SIGNED INTEGER -3
TERMINATE   ;
INT         int
ID         _bar_123
OP_ASSIGNMENT =
SIGNED INTEGER 4
TERMINATE   ;
IF          if
LPAREN (
BOOLEAN STRING true
RPAREN )
LBRACE {
STRING  "Hello World"
TERMINATE ;
RBRACE }
ELSE    else
LBRACE {
STRING  "It's Java Time!"
TERMINATE ;
RBRACE }
INT     int
ID     func
LPAREN (
INT     int
ID     a
RPAREN )
LBRACE {
RETURN return
ID     a
RBRACE }
INT     int
LBRACKET [
RBRACKET ]
ID     a
OP_ASSIGNMENT =
LBRACE {
SIGNED INTEGER 1
SEPARATE ,
SIGNED INTEGER 2
SEPARATE ,
SIGNED INTEGER 3
RBRACE }
TERMINATE ;
CLASS    class

```

	ID        compiler LBRACE { STRING String ID        Teacher TERMINATE ; CHAR char ID        Grade TERMINATE ; RBRACE } SIGNED INTEGER 3 OP_COMPARISON > SIGNED INTEGER 5 SIGNED INTEGER 3 OP_COMPARISON >= SIGNED INTEGER 5 SIGNED INTEGER 3 OP_COMPARISON < SIGNED INTEGER 5 SIGNED INTEGER 3 OP_COMPARISON <= SIGNED INTEGER 5 SIGNED INTEGER 3 OP_COMPARISON != SIGNED INTEGER 5 SIGNED INTEGER 3 OP_COMPARISON == SIGNED INTEGER 5 INT int ID b OP_ASSIGNMENT = SIGNED INTEGER 3 OP_ARITHMATIC * SIGNED INTEGER 5 ID b OP_ASSIGNMENT = SIGNED INTEGER 3 OP_ARITHMATIC / SIGNED INTEGER 5 INT int ID i OP_ASSIGNMENT = SIGNED INTEGER 4 TERMINATE ; WHILE while LPAREN ( ID i OP_COMPARISON < SIGNED INTEGER 5 RPAREN ) LBRACE { ID i OP_ASSIGNMENT = SIGNED INTEGER 10 TERMINATE ; RBRACE } STRING "dddddsdfasdfsdf" ID ghfhhdh Lexical Analyzer Error : Cannot Parse at line 39
--	---