1. Token 정의하기

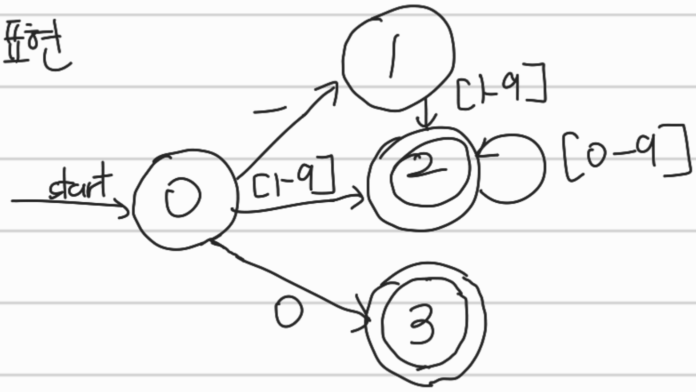
TYPE	int|char|boolean|String
DIGIT	0|1|2|····|9
ALPABET_T	a|b|c|····|z
ALPABET_NT	A|B|C|····|Z
KEYWORLD	if|else|while|class|return|public |static|void|main|args|package
ARITHMATIC_OPERATOR	+|-|*|/
ASSIGNMENT_OPERATOR	=
COMPARISON_OPERATOR	<|>|==|!=|<=|>=
SEMI	;
LBRACE	{
RBRACE	}
LPAREN	(
RPAREN	)
LBRACK	[
RBRACK	]
COMMA	,
WHITESPACE	₩t|₩n|' '

2. 패턴된 Token을 Regular Expression으로 나타내기

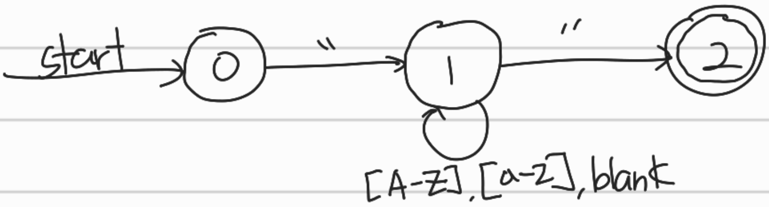
Integer	((-)* [1-9] [0-9]* | 0)
Literal_String	" ([A-Z]* | [a-z]* | [" "] *)* "
Identifier	[a-z]+ [0-9, a-z]* | [-]+ [0-9, a-z]*
Type	int|char|boolean|String
Arithmetic_Operator	+|-|*|/
Assignment_Operator	=
Comparison_Operator	<|>|==|!=|<=|>=
Semi	;
Lbrace	{
Rbrace	}
Lparen	(
Rparen	)
Lbrack	[
Rbrack	]
Comma	,
Whitespace	₩t/₩n/' '

3. DFA 표현

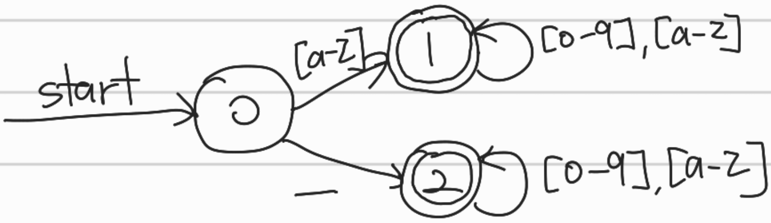
- Integer



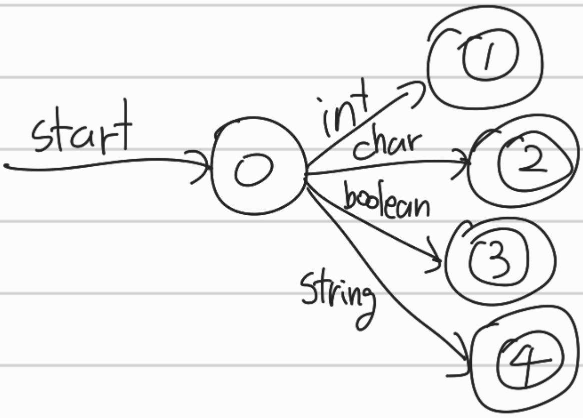
- Literal_String



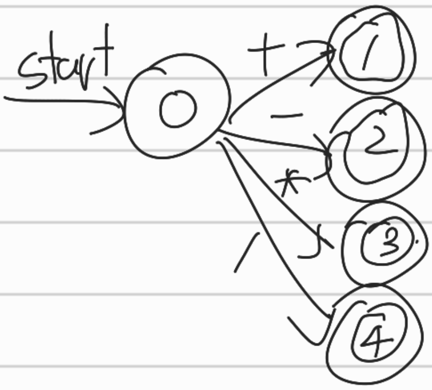
- Identifier



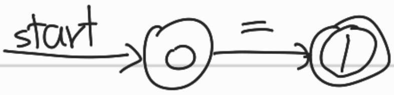
- Type



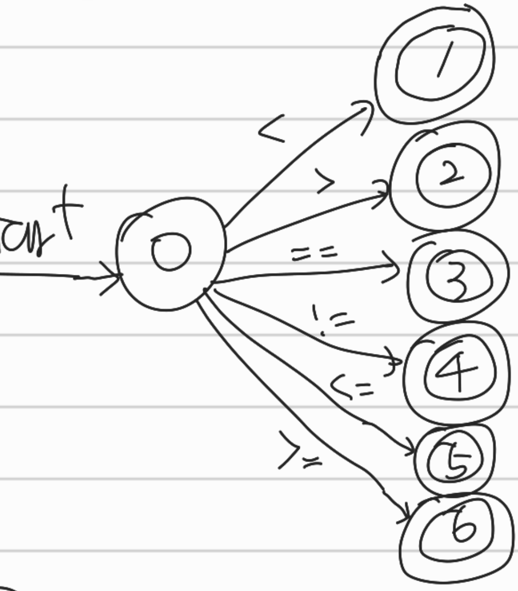
- Arithmetic_Operator



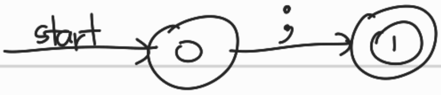
- Assignment_Operator



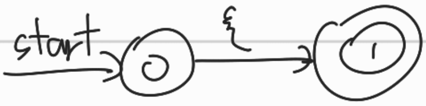
- Comparison_Operator



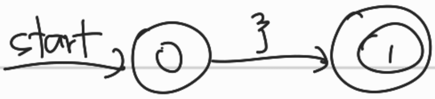
- Semi



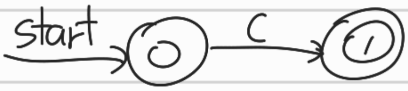
- Lbrace



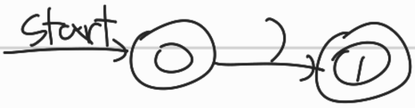
- Rbrace



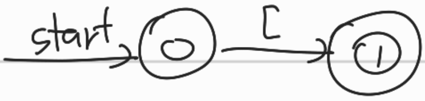
- Lparen



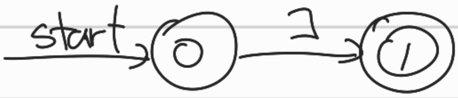
- Rparen



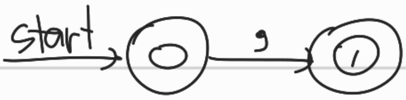
- Lbrack



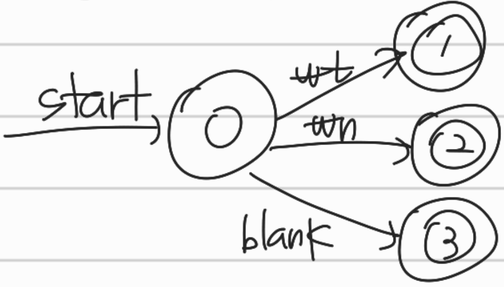
- Rbrack



- Comma



- Whitespace


start
0
\t
\n
blank
1
2
3