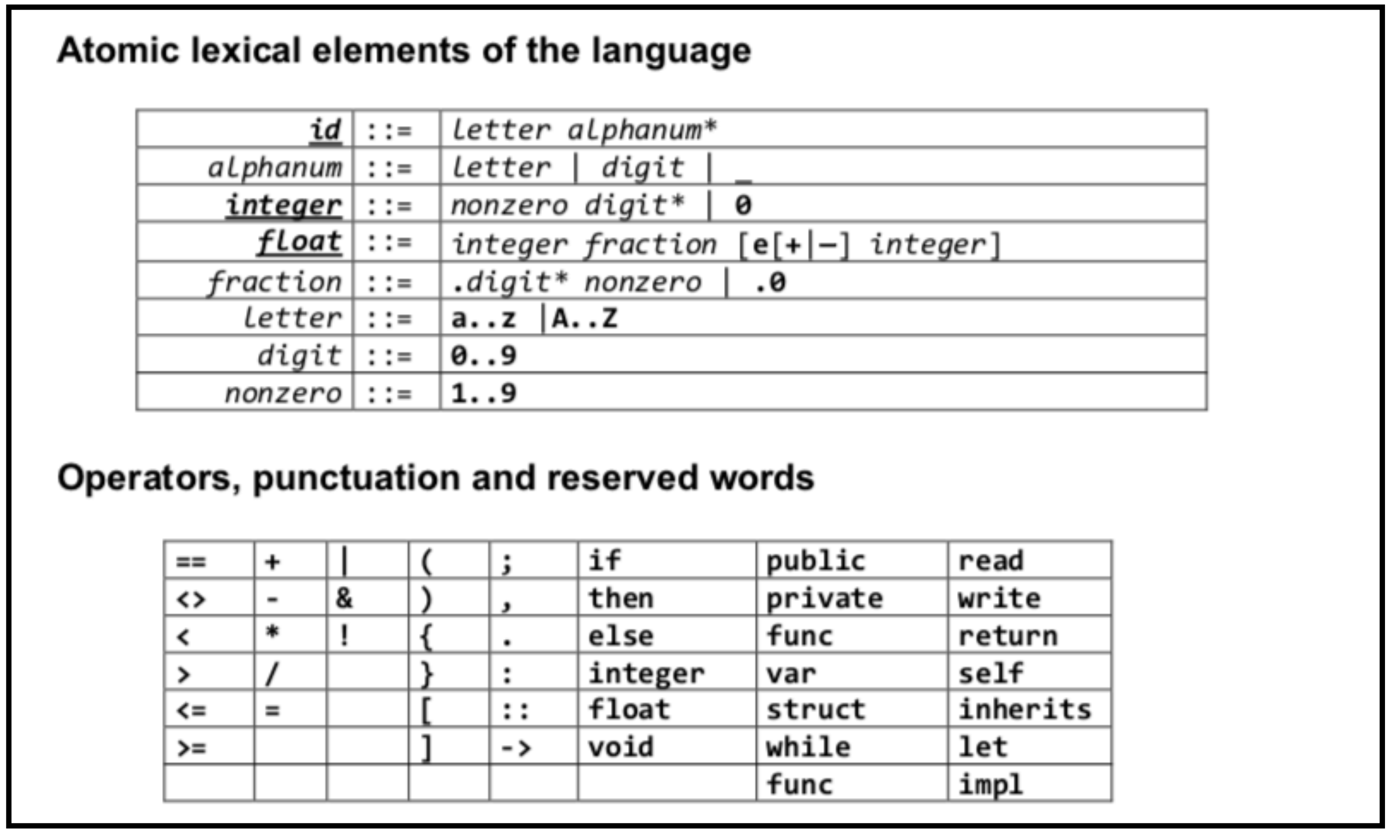
Lexical Analyzer

Compiler Design (COMP 442/6421)
Assignment 1

Name: Garvit Kataria
Student Id: 40155647

Lexical Specifications



Atomic lexical elements of the language

id	::=	*letter alphanum**
alphanum	::=	*letter* | *digit* | _
integer	::=	*nonzero digit** | 0
float	::=	*integer fraction* [e[+|−] *integer*]
fraction	::=	*.digit* nonzero* | .0
letter	::=	a..z |A..Z
digit	::=	0..9
nonzero	::=	1..9

Operators, punctuation and reserved words

==	+	|	(	;	if	public	read
<>	-	&	)	,	then	private	write
<	*	!	{	.	else	func	return
>	/		}	:	integer	var	self
<=	=		[	::	float	struct	inherits
>=			]	->	void	while	let
						func	impl

Atomic lexical elements of the language

id = ((a..z)|(A..Z))((a..z)|(A..Z)|(0…9)|_)*

integer = (1…9)(0…9)*|0

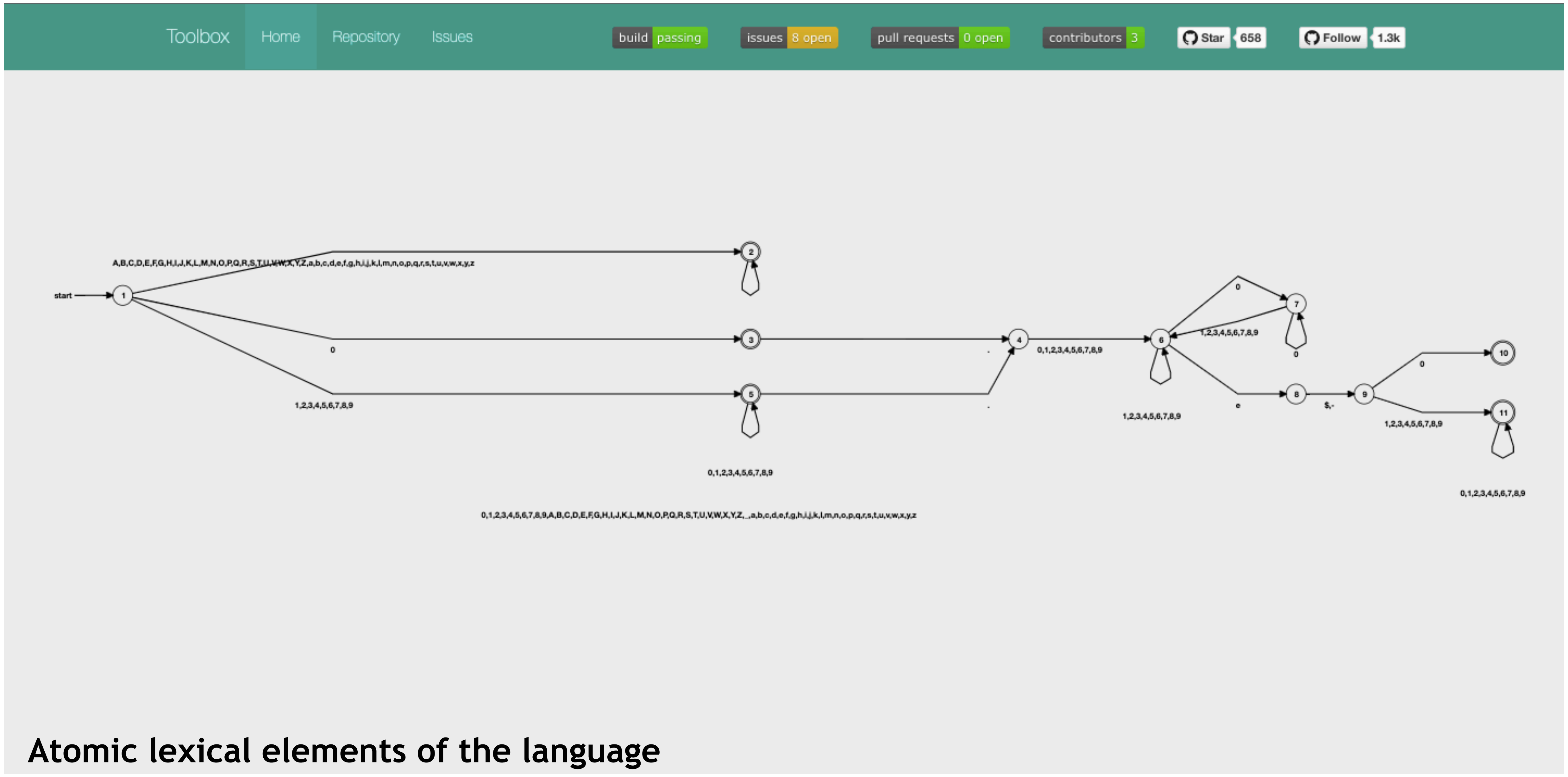
float = ((1-9)(0-9)*|0)((.(0-9)*(1-9))|(.0)) [e[+|-] (1…9)(0…9)*|0]

Operators, punctuation and reserved words

((((((((((((((((((if)|public)|read)|then)|private)|write)|else)|func)|return)|integer)|var)|self)|float)|struct)|inherits)|void)|while)|let)|func)|impl)

(==)|(<=)|(>=)|(=)|(::)|(:)|({)|(})|([)|(])|(,)|(.)|(;)|(&)|(!)|(-)|(x)|(>)|(<)|(/)

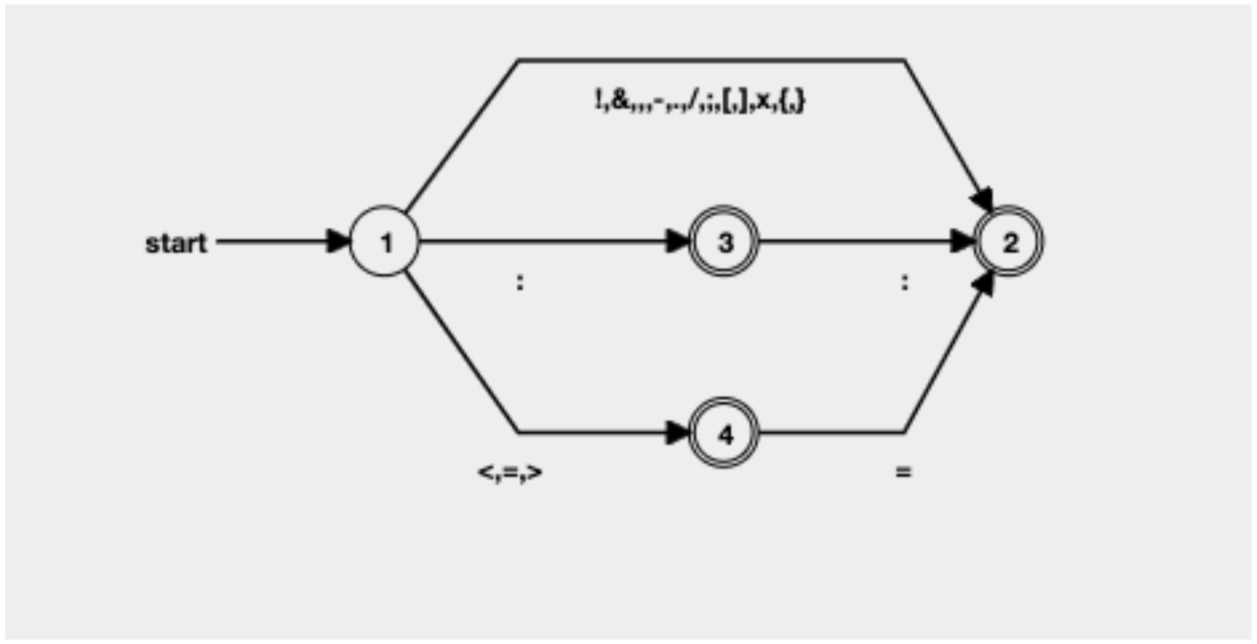
Finite state automaton:



Atomic lexical elements of the language

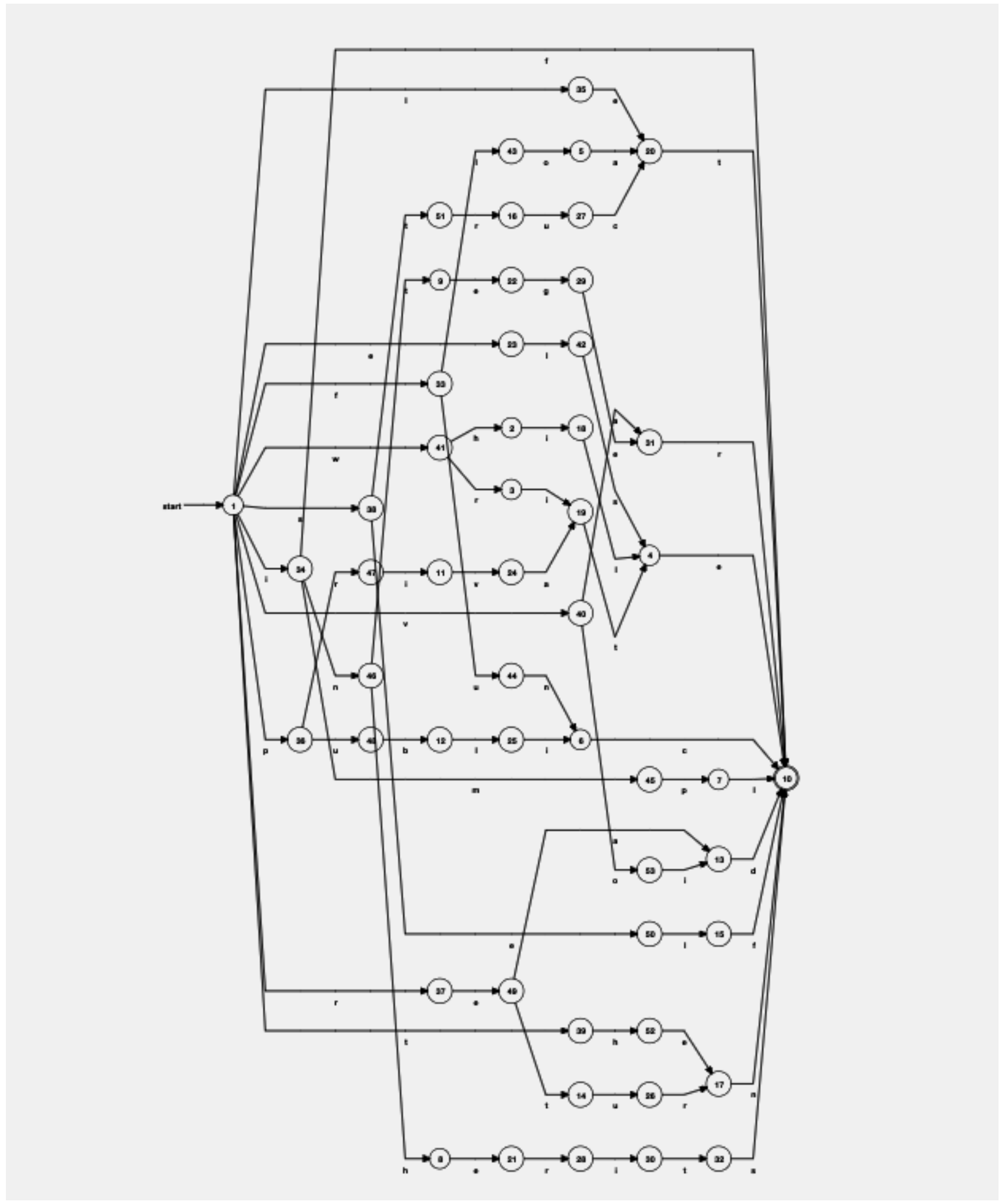
DFA: https://cyberzhg.github.io/toolbox/min_dfa?
regex=KCgoKDF8MnwzfDR8NXw2fDd8OHw5KSgwfDF8MnwzfDR8NXw2fDd8OHw5KSopfDApfCgoKGF8YnxjfGR8ZXxmfGd8aHxpfGp8a3xsfG18bnxvfHB8cXxyfHN8dHx1fHZ8d3x4fHl8eil8KEF8QnxDfER8RXxGfEd8SHxJfEp8S3xMfE18TnxPfFB8UXxSfFN8VHxVfFZ8V3xYfFl8WikpKChhfGJ8Y3xkfGV8ZnxnfGh8aXxqfGt8bHxtfG58b3xwfHF8cnxzfHR8dXx2fHd8eHx5fHopfChBfEJ8Q3xEfEV8RnxHfEh8SXxKfEt8THxNfE58T3xQfFF8UnxTfFR8VXxWfFd8WHxZfFopfCgwfDF8MnwzfDR8NXw2fDd8OHw5KXxfKSopKXwoKCgoMXwyfDN8NHw1fDZ8N3w4fDkpKDB8MXwyfDN8NHw1fDZ8N3w4fDkpKil8MCkoKC4oMHwxfDJ8M3w0fDV8Nnw3fDh8OSkqKDF8MnwzfDR8NXw2fDd8OHw5KSl8LjApKChlKCR8LSkpKCgoMXwyfDN8NHw1fDZ8N3w4fDkpKDB8MXwyfDN8NHw1fDZ8N3w4fDkpKil8MCkpKQ==

Operators, punctuation



DFA: https://cyberzhg.github.io/toolbox/min_dfa?regex=KD09KXwoPD0pfCg+PSl8KD0pfCg6Oil8KDopfCh7KXwofSl8KFspfChdKXwoLCl8KC4pfCg7KXwoJil8KCEpfCgtKXwoeCl8KD4pfCg8KXwoLyk=

Reserved Words



https://cyberzhg.github.io/toolbox/min_dfa?
regex=KCgoKCgoKCgoKCgoKCgoKCgoKChpZil8cHVibGljKXxyZWFkKXx0aGVuKXxw
cml2YXRlKXx3cml0ZSl8ZWxzZSl8ZnVuYyl8cmV0dXJuKXxpbnRlZ2VyKXx2YXIpfHNlbG
YpfGZsb2F0KXxzdHJ1Y3QpfGluaGVyaXRzKXx2b2lkKXx3aGlsZSl8bGV0KXxmdW5jKX
xpbXBsKQ==

Design

The scan() method in the lexical analyser consists of 3 parts, these parts are the DFAs shown in this slide in finite state automaton section.
1. Handles all the **Operators** & **punctuation**.
2. Handles **Reserved Words** & **Id.**
3. Handles **Integer** and **Floats**.

Part 1: **Operators & punctuation**
This is a bunch of case statements which figure out the token as the lexer reads the next character from the file. All the comments inline, block & nested comments are handled in this section.

Part 2: **Handles Reserved Words & Id**
According to the DFA , All the reserved words are stored in a hash map so when the lexer process an Id it checks for the reserved words with a constant time lookup.

Part 3: **Handles Integer and Floats.**
This is according to the DFA.

Use of tools

I have used an open source tool to convert simple regular expressions to minimum deterministic finite automaton.
Link -> https://cyberzhg.github.io/toolbox/min_dfa

float = (((1|2|3|4|5|6|7|8|9)(0|1|2|3|4|5|6|7|8|9)*)|0)((.(0|1|2|3|4|5|6|7|8|9)*(1|2|3|4|5|6|7|8|9))|.0)((e($|-))(((1|2|3|4|5|6|7|8|9)(0|1|2|3|4|5|6|7|8|9)*)|0))
here $ is +sign

integer = ((1|2|3|4|5|6|7|8|9)(0|1|2|3|4|5|6|7|8|9)*)|0

id = ((a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z)|(A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z))((a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z)|(A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z)|(0|1|2|3|4|5|6|7|8|9)|_)*

Regular Expression for the Grammer -> ((((1|2|3|4|5|6|7|8|9)(0|1|2|3|4|5|6|7|8|9)*)|0)|(((a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z)|(A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z))((a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z)|(A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z)|(0|1|2|3|4|5|6|7|8|9)|_)*))|((((1|2|3|4|5|6|7|8|9)(0|1|2|3|4|5|6|7|8|9)*)|0)((.(0|1|2|3|4|5|6|7|8|9)*(1|2|3|4|5|6|7|8|9))|.0)((e($|-))(((1|2|3|4|5|6|7|8|9)(0|1|2|3|4|5|6|7|8|9)*)|0)))