Lab Assignment 2- Lexical Analysis

In this assignment, you will work on implementing a lexical analyzer in your preferred programming language.

We will consider some basic constructs (e.g., if-else statements), **understand what are the different tokens (and patterns for the tokens)** in the considered language/constructs.

Implement a lexical analyzer (using any programming language) for the considered tokens/patterns. Your program should take a statement as input and return the sequence of tokens as output.

Let us consider the example grammar for branching statements given below:

stmt	→	**if** *expr* **then** *stmt*
	|	**if** *expr* **then** *stmt* **else** *stmt*
	|	ϵ
expr	→	*term* **relop** *term*
	|	*term*
term	→	**id**
	|	**number**

The patterns for the tokens in the language are described below:

digit	→	[0-9]
digits	→	$digit^+$
number	→	*digits* (. *digits*)? (E [+-]? *digits*)?
letter	→	[A-Za-z]
id	→	*letter* (*letter* | *digit*)*
if	→	`if`
then	→	`then`
else	→	`else`
relop	→	< | > | <= | >= | = | <>

Implement a lexical analyzer using any programming language. Your program should take a statement as input and return the sequence of tokens as output.

- **Example Input**: `if input<10 then output1=100 else output2>=100`
- **Expected output**:

```
(if,)
(id,input)
(relop,<)
(number,10)
(then,)
(id,output1)
(relop,=)
(number,100)
(else,)
(id,output2)
(relop,>=)
(number,100)
```
