FAQ

What should I do and implement for term-project #1??

1. **(By hand)** Define tokens based on the given lexical specifications
2. **(By hand)** Make a regular expression for each token
3. **(By hand)** Construct a NFA for recognizing tokens
4. **(By hand)** Construct a DFA transition table for recognizing tokens
5. Implement a program which lexically analyzes an input source code by using the DFA transition table

Can we use special libraries or programs that help lexical analysis?

- No!!

  Ex1) if you use Pattern and Matcher classes in Java, you will earn no points

  Ex2) Regex library

FAQ

Should I use the rules we've studied for constructing NFA and DFA??

No, it's not mandatory (only for the project, not for exams)

All the decisions related to your implementation is up to you

- You can freely make NFA and DFA without following the rules
- If you want, you can directly construct DFA from regular expressions
- You can use a single huge DFA for checking $if\ substring \in L(merged)$ or

  a multiple small DFA for checking $if\ substring \in L(id)||substring \in L(vtype)||$ …

But, your program must work correctly (your DFA must correctly recognize tokens)

If you cannot ensure the correctness of your DFA, it would be better to follow the rules

How to define tokens??

As you want!!!!! It's up to you

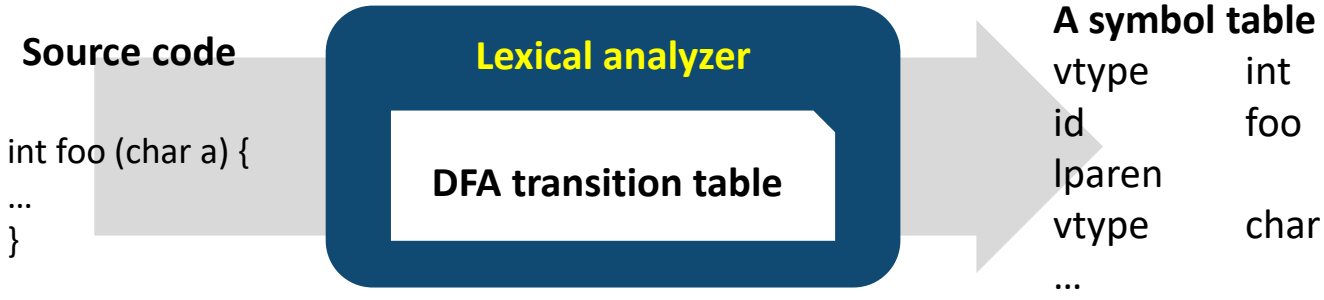
e.g., for variable types, you can make two different tokens int and char

or an integrated token vtype with an additional attribute

FAQ

What are the inputs and outputs of a lexical analyzer?

- Input: a source code written in simplified Java programming languages
- Output: a symbol table which contains information about tokens (e.g., token names and attributes)



Should we consider the syntactic grammars?

- No. It will be checked in a syntax analyzer