Checkin 9

Calculate FIRST sets for the below grammar

S ::= **lpar** X **rpar**
X ::= **id comma** X
| ε

Administrivia

Administrivia

- The social contract

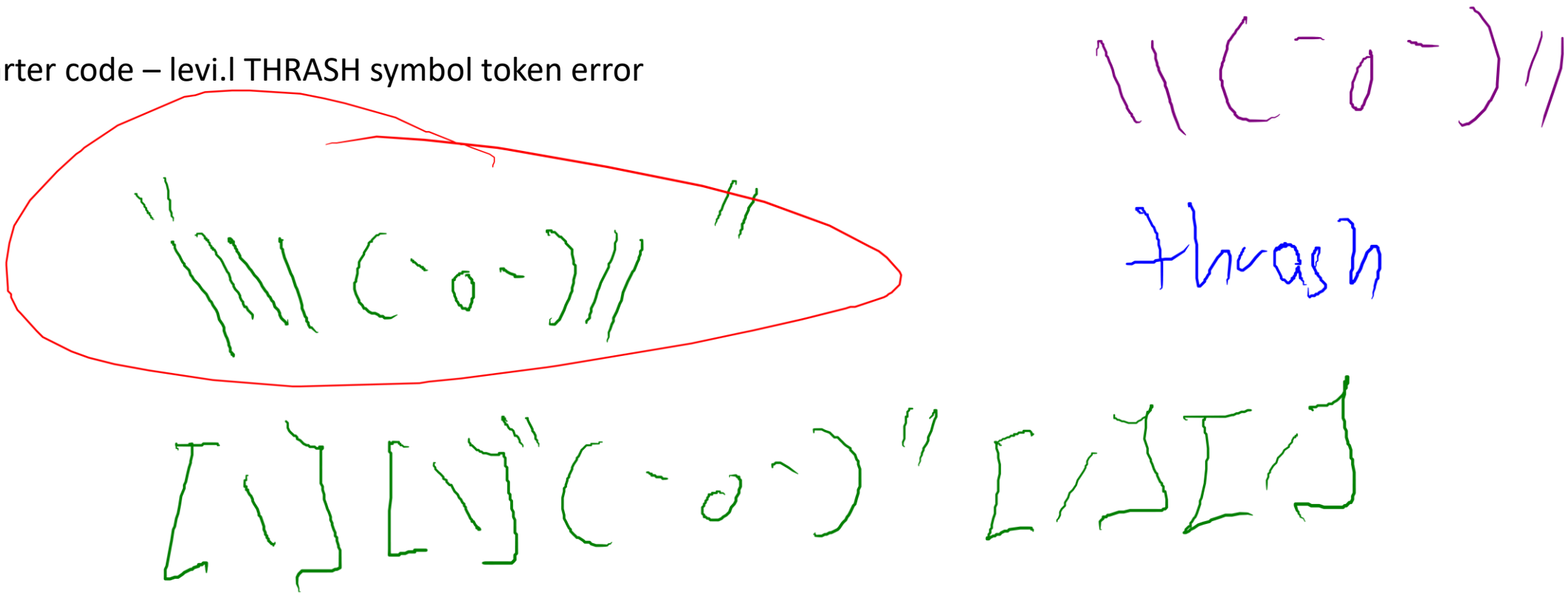
Administrivia

P1 Graded

Flipped Wednesday

Errata

P2 starter code – levi.l THRASH symbol token error



Checkin 8 - Redux

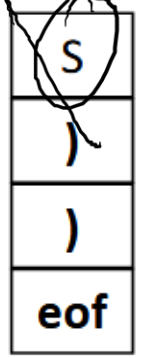
Whoops!

Assume an LL(1) parser with…

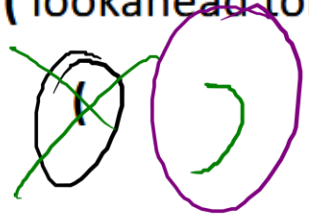
this selector table:

	(	)	{	}
S	(S)	)	{	}

this syntax stack:



and this **(** lookahead token:

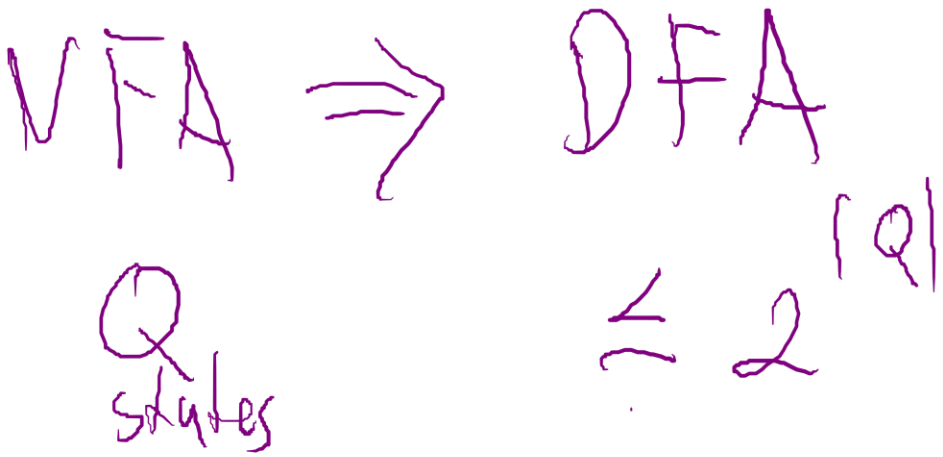


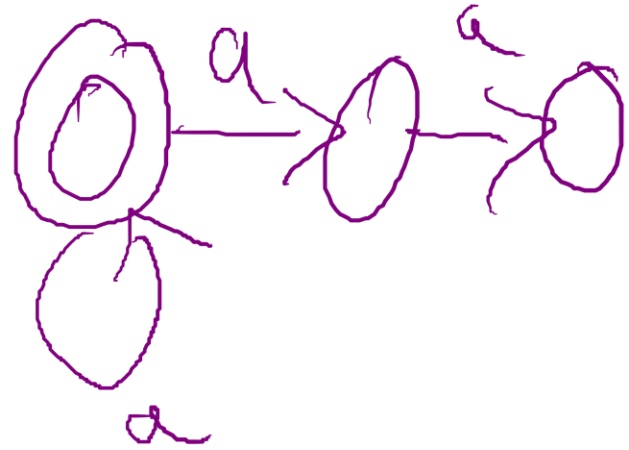
Draw the configuration of the parser after it processes the tokens **( )**
assume the next character thereafter is an eof

Quiz #1 – Question 1, Part I

Assume, for this question, that you are the captain of a pirate ship.

Your first mate claims that there exists an NFA with 3 states, but that there exists an equivalent DFA with only 1 state. Is your first mate correct? If so, give an example of such an NFA / DFA pair. If not, explain why no pair exists.





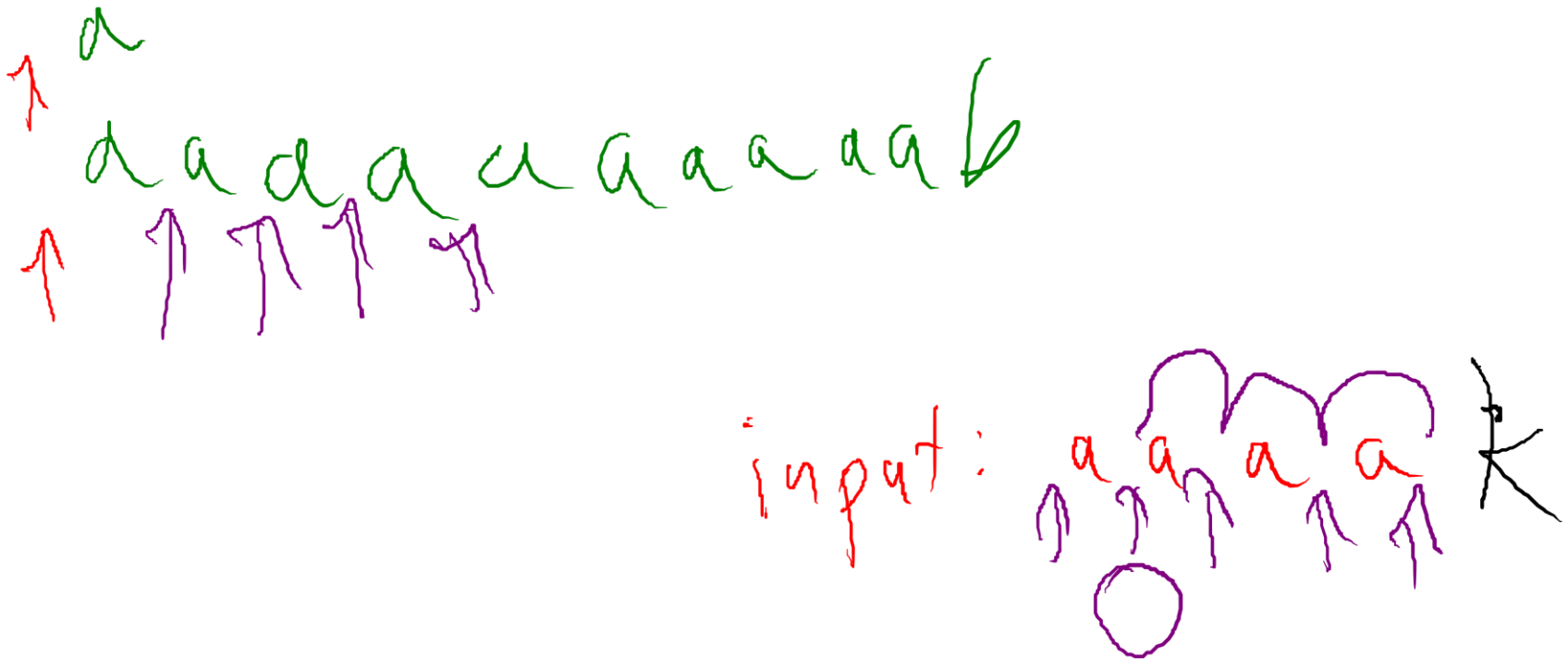
Quiz #1 – Question 1, Part II

Assume, for this question, that you are the captain of a pirate ship.

Your helmsman claims that there exists an NFA with 3 states, and that any equivalent DFA has at least 8 states. Is your helmsman correct? If so, explain why. If not, give a counterexample NFA / DFA pair.

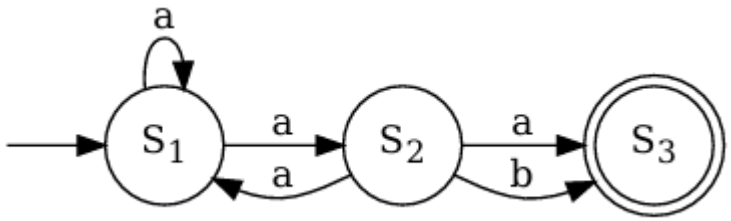
Quiz #1 – Question 2

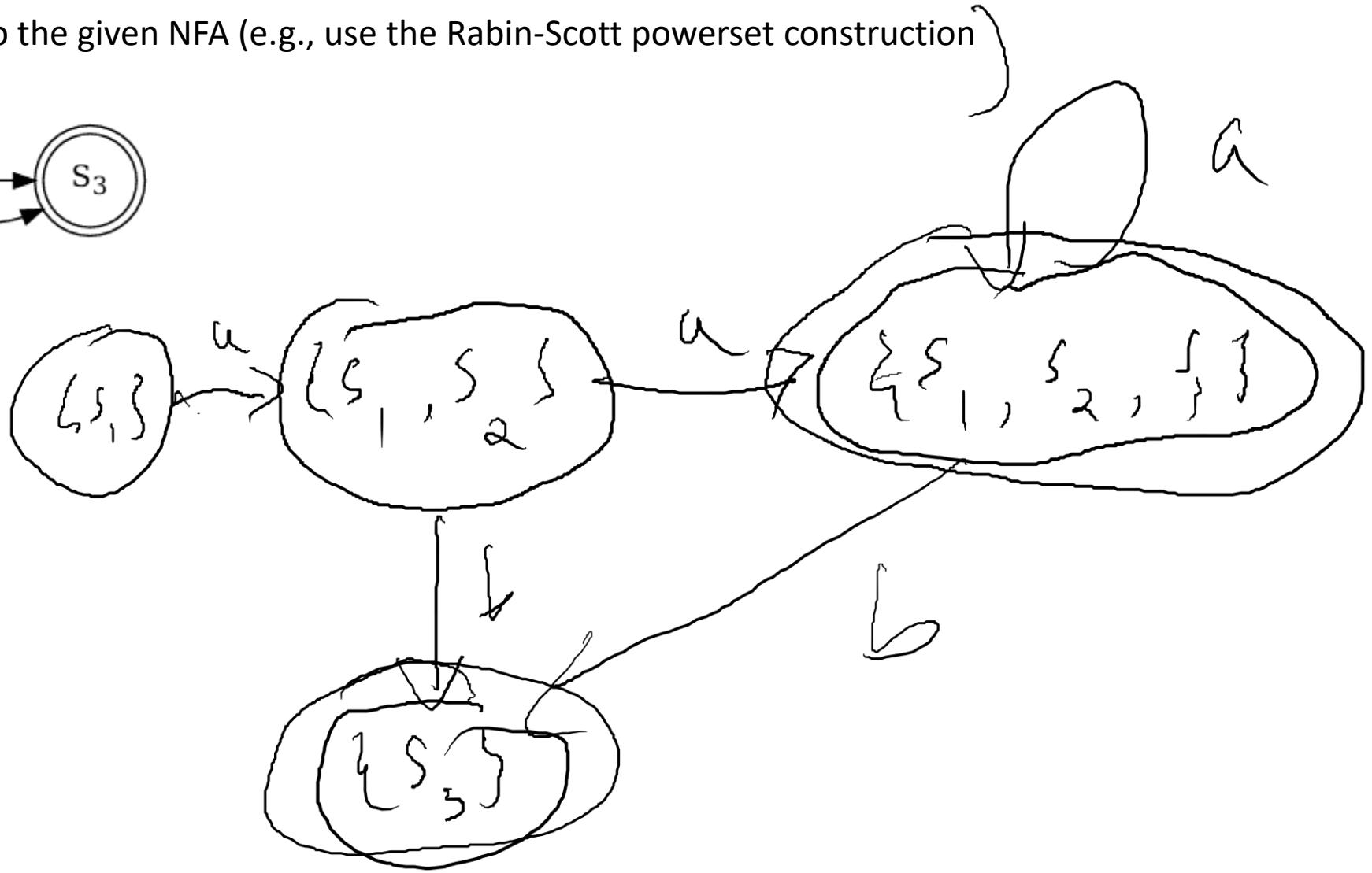
We described a method by which DFAs could be used to create a tokenizer (i.e. a translator from a stream to characters to a stream of tokens). The tokenizer had to backtrack over the input string even after an accepting state had been been found for one of the token DFAs. Write a series of token languages and an input string that would cause the tokenizer to backtrace by at least 2 characters. Explain why this backtracking happens.



Quiz #1 – Question 3

Create an equivalent DFA to the given NFA (e.g., use the Rabin-Scott powerset construction



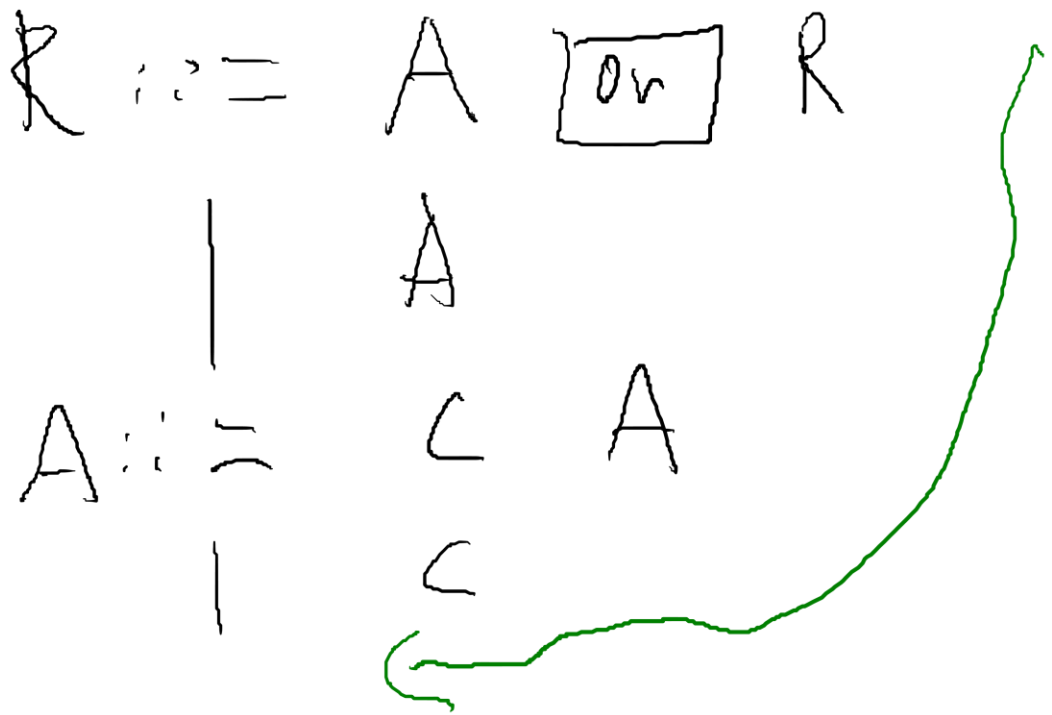


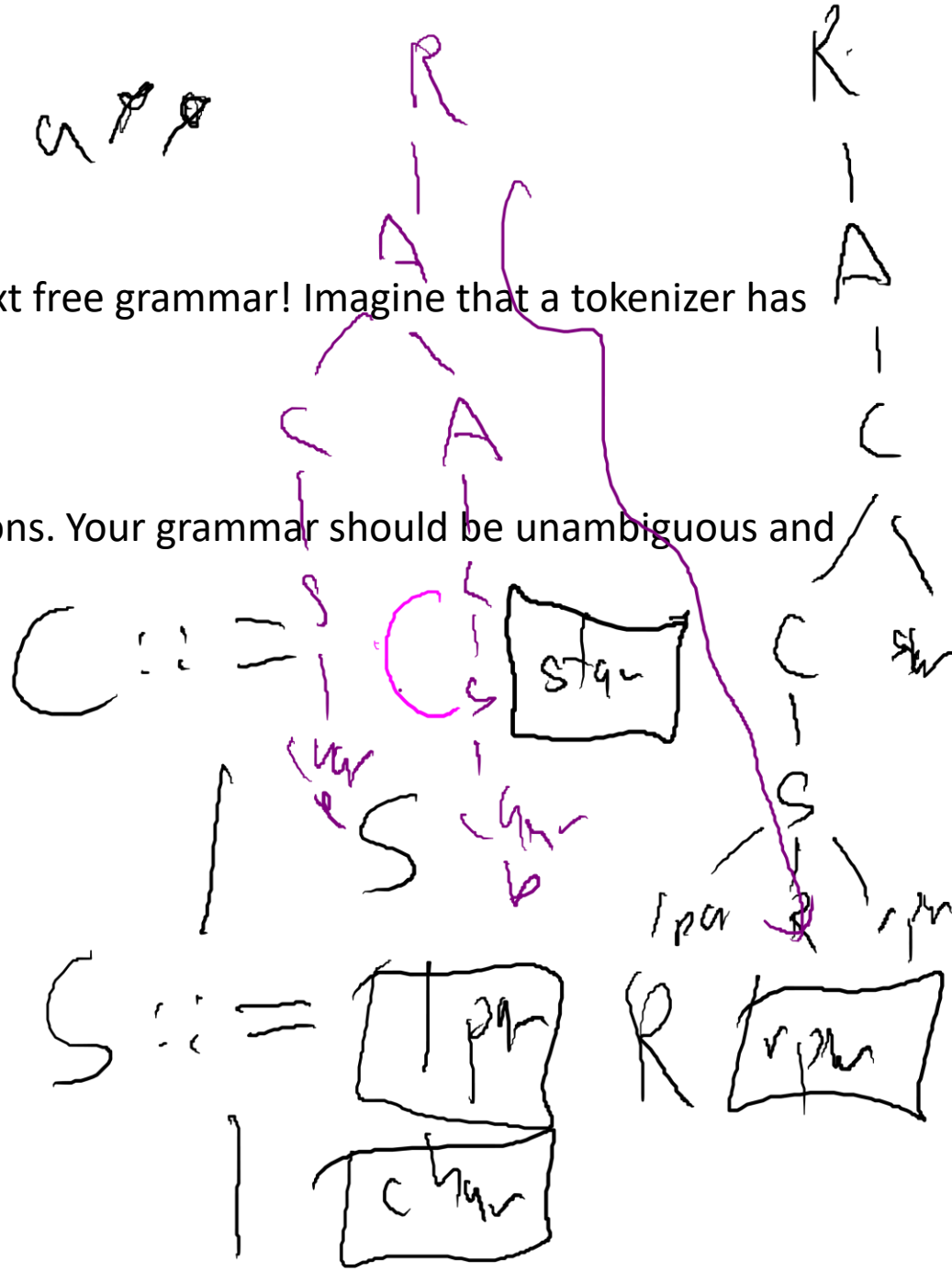
Quiz #1 – Question 4

The language of all regular expressions can itself be expressed as a context free grammar! Imagine that a tokenizer has already built that uses the tokens

star	or	lpar	rpar	char

Write a context-free grammar for token streams of valid regular expressions. Your grammar should be unambiguous and respect the precedence of the operators.





Quiz #1 – Question 5

Imagine the following is the rules section of a Flex spec:

```
by              {std::cout << "1\n"; }
cr(o+)[^m]      {std::cout << "2\n"; }
[.\n]+          {std::cout << "3\n"; }
...             {std::cout << "4\n"; }
```

What does this spec print on the following input: `bycrom`?