1. In Java, we can print to console without a main method using a static block in a class if JDK version <= 1.6, otherwise an exception is thrown as no main methods is defined.

2. Indentation is a reason why some language like Python is not regular. Having to compare indentations of consecutive lines makes regular grammar not working!

3. In a grammar, rule produce strings, so they are called **productions**. Each production has a **name** and a **body** (body is simply a list of symbols):

a. A terminal is a letter from alphabet as an end point as they don't lead to any further moves.

b. A nonterminal is a named reference to another rule in the grammar. It means play the rule and insert whatever it produces here.

c. I can have multiple rules with the same name.

4. An example of grammar rules:



5. The **Visitor Pattern**: It makes it easy to add a new operation to a bunch of types. Create an interface “***ObjectVisitor***” which holds functions to visit each of the types. When I want a new operation, I simply create a new class implementing the visitor interface. I also add an abstract function ***accept(ObjectVisitor visitor)*** in the super class of these types. For each of the types, I implement the accept function to make it call the corresponding function of itself in the visitor. When I want type TA to do operation OA, I simply call ***TA.accept(OA)***.

6: **Parsing techniques**: many, including parser combinators, Earley parsers, the shunting yard algorithm, and packrat parsing, but we use **Recursive Descent** in jlox. It is a top-down parser as it starts from the outermost (***expression*** in this case) production to the innermost (***primary*** in this case).

Need to set up a grammar rule without **ambiguity** as the work of parser is to find out from a string what production rule it corresponds to. So, we also need to set up precedence and associativity.

7. **Synchronization** when encounters parsing error: s soon as the parser detects an error, it enters panic mode. It knows at least one token doesn’t make sense given its current state in the middle of some stack of grammar productions.

Before it can get back to parsing, it needs to get its state and the sequence of forthcoming tokens aligned such that the next token does match the rule being parsed.本质上就是将出错的statement的其他tokens丢掉，直到下一句statement。