***Lecture 1 – Lexical Analysis***

**Compiler** – translates ‘Source language’ to ‘Target language’.

**Error checking -**

**Lexeme** – a series of chars separated by a convention

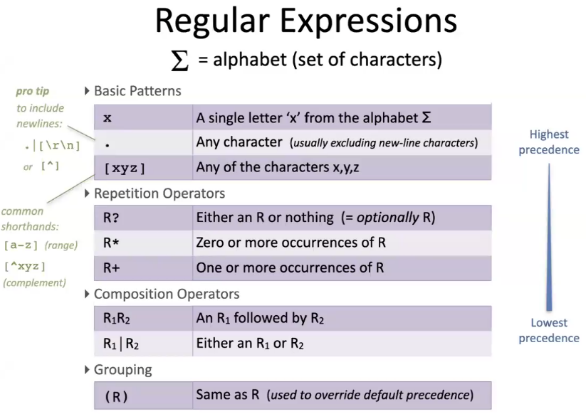
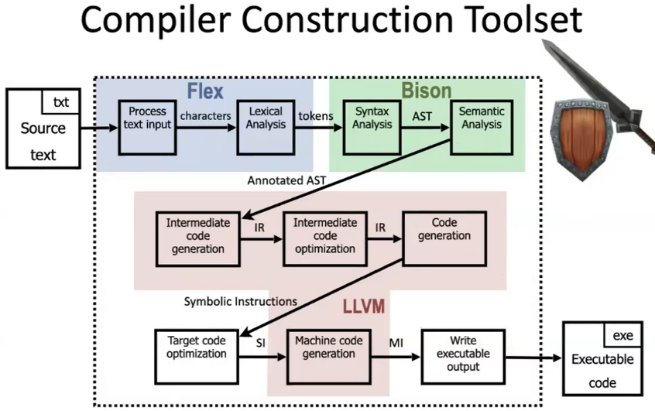
**Pattern -**  a rule specifying a set of strings

**Token –** <kind, attributes>

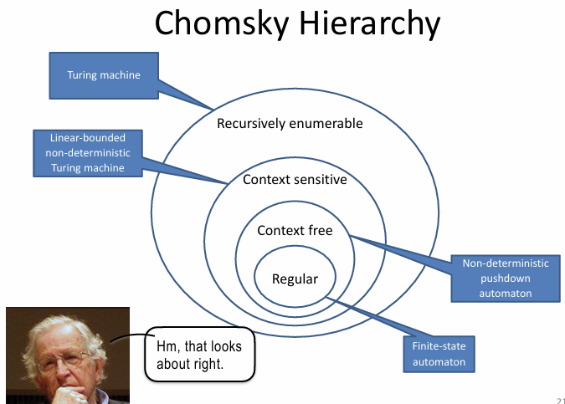
Ambiguity solution – find longest matching token

Conflict Resolution:

1. Pick longest lexeme
2. Pick Token with higher priority(first in the definitions file)



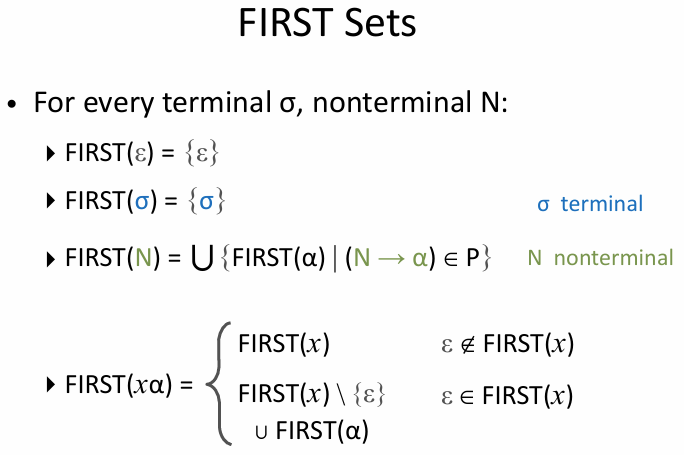
L2 – P31-34 – in which file should this code be?

***Lecture 2 – Syntax Analysis***

***V –*** *non-terminals****; T –*** *Terminals(Tokens);* ***P –*** *Derivation Rules;* ***S –*** *Start Symbol*

***Language –*** *a set of strings of terminals derivable from the start symbol*

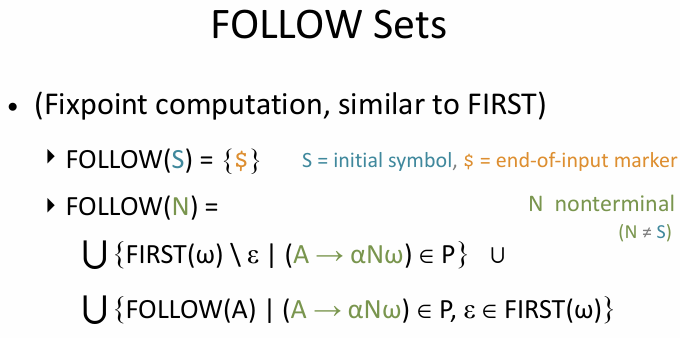
***Top-down Parser***

* *Construct Leftmost derivation*
* *Apply rules from left to right*
* *Predict what rule to apply based on nonterminal and token*

***LL(K) Grammars –***

* *Top-Down analysis*
* *Scan input from left to right*
* *Producing the leftmost derivation*
* *Lookahead of k tokens*

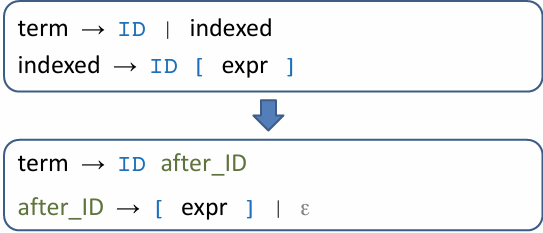
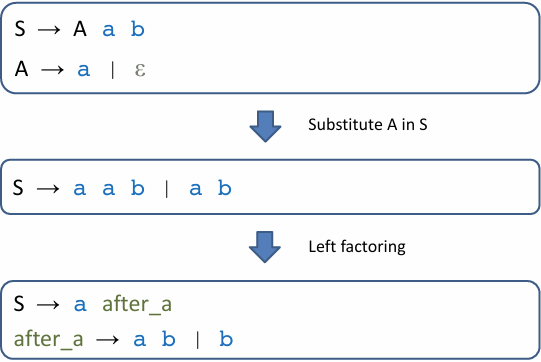
*****LL(1)*** *grammar exists there are no conflicts*

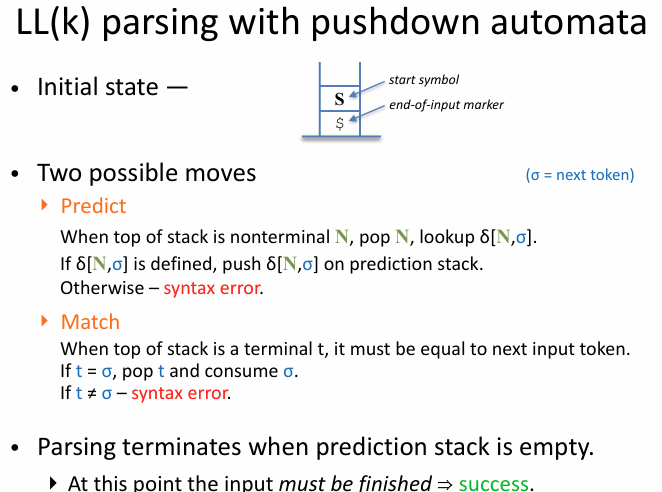
***First(a) –*** *all terminals that ‘a’ can start with*

*there exists some derivation sequence*

**Follow(a)**  **-**  all terminals that may show up right after ‘a’

***LL(1) conflicts***

* ***First/First conflict –*** *Solved using Left Factoring*
* ***First/Follow conflict –*** *solved using Grammatical Substitution*



* ***A blue arrow pointing to the right

  Description automatically generatedLeft Recursion –*** *Left Recursion Removal*

***A close up of a word

Description automatically generated***

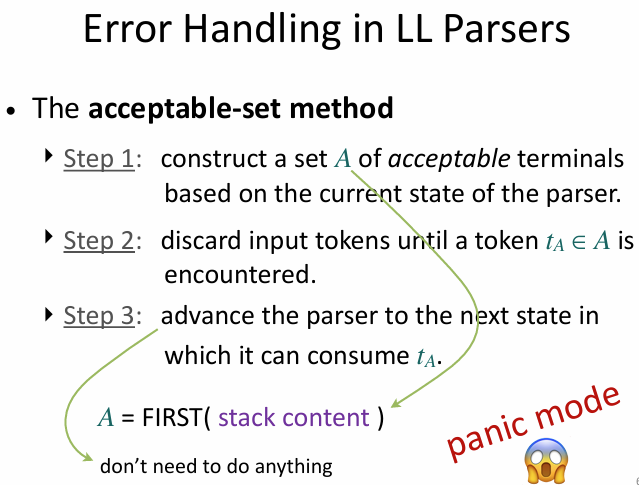
***A white background with black text and numbers

Description automatically generated***

******

***A white background with black text

Description automatically generated***

******

***A screenshot of a computer

Description automatically generated***

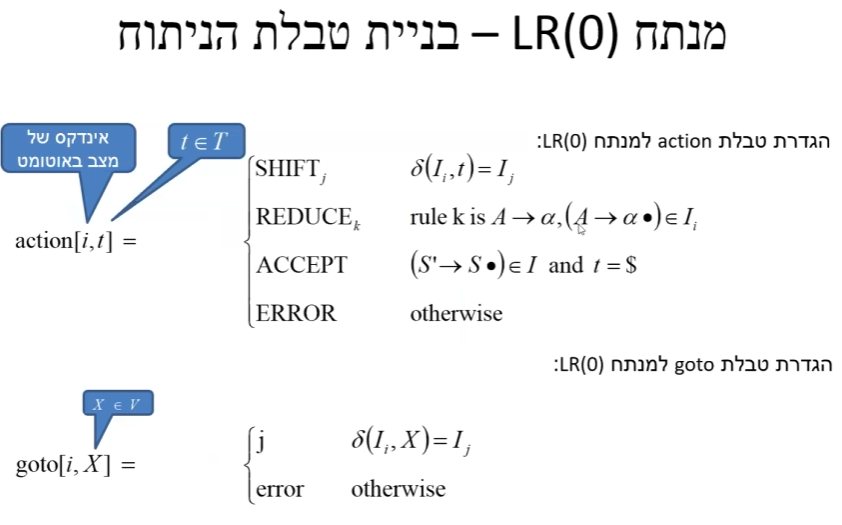
***A text on a white background

Description automatically generated***

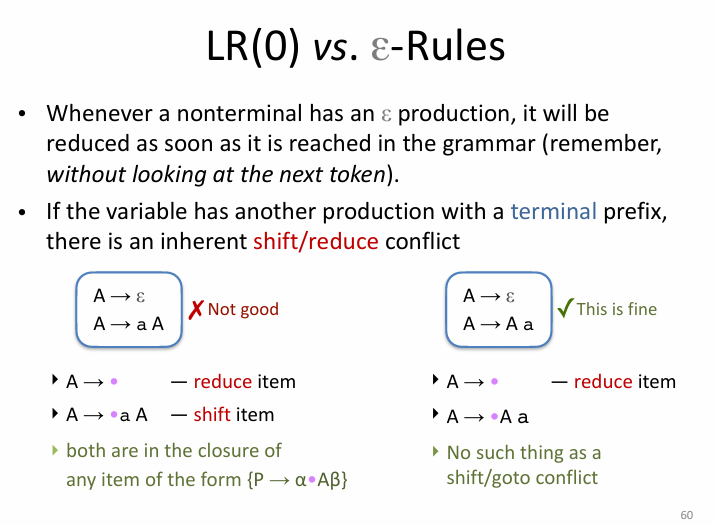
***Lecture 3 – Syntax Analysis – Parsing***

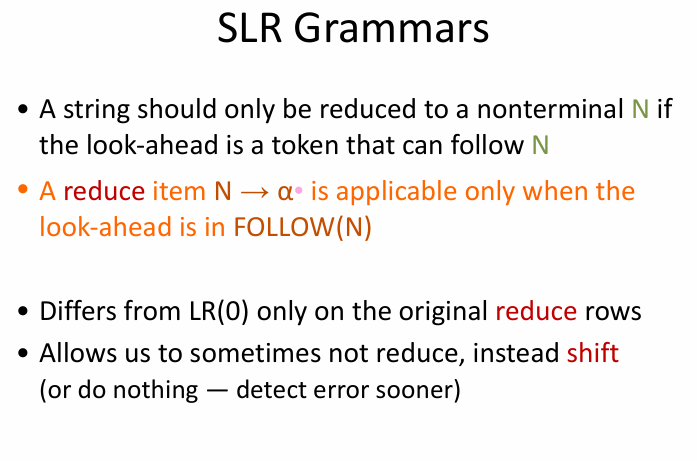
***Bottom-up Parser***

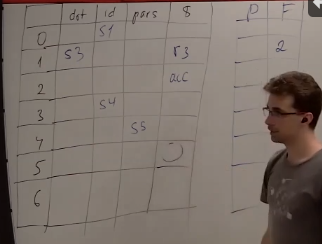
* *Construct rightmost derivation*
* *Apply rules from right to left*
* *Reduce a right-hand side of a production to its non-terminal*

***LR(K) Grammars –***

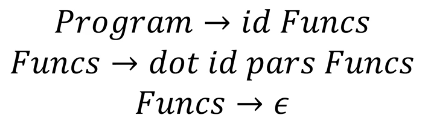
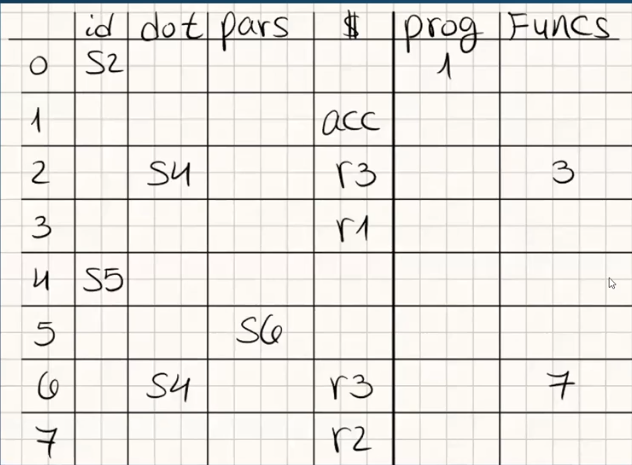
* *Bottom-Up analysis*
* *Scan input from left to right*
* *Producing the rightmost derivation*
* *Lookahead of k tokens*

****

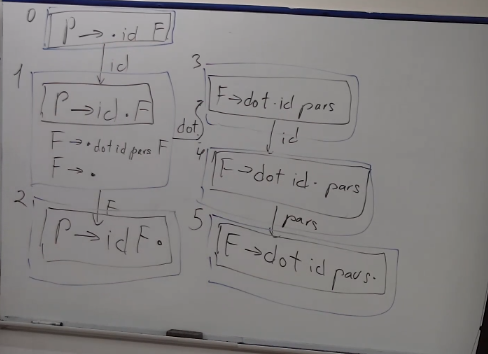
****

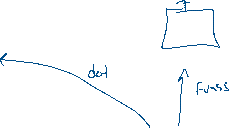
****

Every language in LR(0) is also in SLR

****



****



**Semantic analysis**

**Runtime analysis**