**CFG Parser**

**Team Members’ Names and IDs**

* Sama Eldessouky – 221000579
* Ziad Wael – 221001480
* Nouran Ahmed – 221000767
* Aly Mohamed Sakr – 221002050
* Shaden Abdelsalam – 221001937

**Detailed Description**

**What does it do?**  
This project is a program that checks whether a given string can be generated by a context-free grammar (CFG) defined by the user. It validates the input string against the grammar rules and, if the string is valid, produces a visual parse tree showing how the string corresponds to the grammar’s structure. This visualization helps users understand the syntax and parsing process.

**Input Format**  
The user inputs grammar rules one by one, each following the format:  
NonTerminal -> Production1 | Production2 | ...  
Each production is a sequence of symbols separated by spaces. After entering all rules, the user types done. Then, the user enters the string to be parsed, with tokens separated by spaces.

**Output Format**  
The program outputs whether the input string is valid or invalid according to the grammar. If valid, it generates a parse tree image (parse\_tree.png) that visually represents the derivation of the string from the grammar.

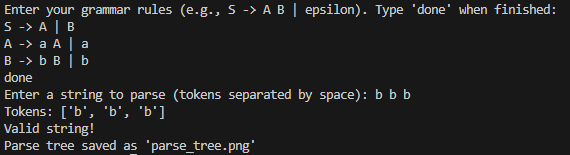
**Inside Mechanism**  
The program uses a recursive parsing algorithm with memoization to efficiently check all possible derivations for the input string. It builds a parse tree structure while parsing, which is later rendered into an image using the Graphviz library.

**Programming Language, Tools & Libraries Used**

* **Programming Language:** Python 3
* **Libraries:**
  + graphviz for generating parse tree images
* **Tools:**
  + Python IDE or editor
  + Graphviz software (required for rendering the images)

**Images of the output**

* **input**



* **output**

A diagram of a diagram

AI-generated content may be incorrect.