https://github.com/cs-ubbcluj-ro/lab-work-computer-science-2024-dragosgavrus1/tree/main/3-Parser

Gavrus Dragos Andrei

Class: Grammar

Purpose

Represents a formal grammar defined by:

- **Non-terminals (N)**
- **Terminals (E)**
- **Start symbol (S)**
- **Productions (P)**

Methods

__init__(N, E, S, P)

Initializes the grammar with:

- **N (set): Non-terminal symbols.**
- **E (set): Terminal symbols.**
- **S (str): Start symbol.**
- **P (dict): Productions, mapping non-terminals to a list of right-hand side (RHS) alternatives.**

---

from_file(filename)

Reads a grammar definition from a file and validates it.

- **File format:**
  - **Line 1: Non-terminals (N = <space-separated list>)**
  - **Line 2: Terminals (E = <space-separated list>)**
  - **Line 3: Start symbol (S = <symbol>)**
  - **Line 4: Empty**
  - **Lines 5+: Productions (<LHS> -> <RHS> | ...)**
- **Returns: A valid Grammar instance or an error message if validation fails.**

---

parse_line(line)

Parses a single line of the grammar definition to extract non-terminals or terminals.

- **Parameters: line (str) – A line of text in key = value format.**
- **Returns: List of symbols (str).**

---

parse_productions(lines)

Parses production rules from multiple lines.

- **Parameters: lines (list of str) – Each line contains a production rule (LHS -> RHS).**
- **Returns: Dictionary mapping non-terminals to RHS alternatives.**

---

validate(N, E, S, P)

Validates the grammar to ensure:

- **The start symbol (S) is in the non-terminals (N).**
- **Production rules (P) only contain symbols from non-terminals (N) and terminals (E).**
- **Returns: True if valid, False otherwise.**

---

is_cfg()

Checks if the grammar is a context-free grammar (CFG).

- **Returns: True if the grammar is CFG, False otherwise.**

---

get_nonterminal_productions(nonterminal)

Fetches the productions for a given non-terminal.

- **Parameters: nonterminal (str) – The non-terminal symbol.**
- **Returns: List of RHS alternatives.**
- **Raises: Exception if the symbol is not a non-terminal.**

__str__()

String representation of the grammar, showing its components (N, E, S, P).