https://github.com/alexreitler/FLCD_Lab5/tree/output

The `Output` class is responsible for constructing and representing the parse tree structure based on the output of a parser. It facilitates the visualization of the hierarchical relationships between nodes in the parse tree. The class utilizes lists and maps to store information about nodes, their parent-child relationships, and the number of children for each node type.

Constructor

- `Output(grammar: Grammar)`: Initializes an `Output` instance with a given grammar. The constructor sets up data structures to store information about nodes, parent-child relationships, and the number of children for nonterminals, terminals, and the epsilon symbol.

Properties

- `grammar: Grammar`: The grammar associated with the parser output.
- `nrChildren: MutableMap<String, Int>`: A map storing the number of children for each nonterminal, terminal, or epsilon symbol.
- `values: MutableList<String>`: A list storing the values of nodes in the parse tree.
- `father: MutableList<Int>`: A list storing the indices of parent nodes.
- `leftChild: MutableList<Int>`: A list storing the indices of left children.
- `rightSibling: MutableList<Int>`: A list storing the indices of right siblings.

Public Methods

1. `addProductionString(productionString: List<String>)`:
    - **Parameters:** `productionString` - A list of strings representing a production sequence.
    - **Description:** Adds a production sequence to the parse tree structure.
2. `add(node: String, parent: Int): Int`:
    - **Parameters:**
        - `node` - A string representing a node in the parse tree.
        - `parent` - An integer representing the index of the parent node in the parse tree.
    - **Returns:** The index of the added node in the values list.
    - **Description:** Adds a node to the parse tree with the specified parent.
3. `toString(): String`:
    - **Returns:** A string representation of the parse tree.
    - **Description:** Generates a string representation of the parse tree, including values, fathers, left children, and right siblings. The string is suitable for debugging and understanding the structure of the parse tree.
4. `subtree(node: Int): String` (Private Method):
    - **Parameters:** `node` - The index of the current node in the parse tree.
    - **Returns:** A string representation of the subtree rooted at the given node.
    - **Description:** Recursively constructs a subtree representation, including proper indentation and linking symbols.