Documentation

https://github.com/iliedr19/lftc34

The lexical analyzer processes the input code and identifies tokens such as keywords, identifiers, operators, separators, and constants.

Features

- Tokenization: Breaks down the input code into meaningful tokens.
- Token Types:
  - Keywords: Reserved words in the programming language (e.g., `if`, `else`, `while`).
  - Identifiers: Names of variables, functions, etc.
  - Operators: Mathematical and logical operators (`+`, `-`, `*`, `==`, `!=`, `<`, `>`).
  - Separators: Characters that separate tokens (`()`, `{}`, `;`).
  - Constants: Numeric or string values (`123`, `"string"`, `'char'`).
- Error Handling: Detects and reports lexical errors like invalid tokens or unrecognized characters.

Components

1. MyScanner Class: Contains methods to read and process input code, tokenize it, and detect lexical errors.
   - `read_file()`: Reads the input code from a file.
   - `create_list_of_program_elems()`: Breaks down the code into a list of tokens.
   - `scan()`: Identifies and categorizes tokens and reports lexical errors.
   - `get_pif()`: Retrieves the Program Internal Form (PIF) - a data structure storing token information.
   - `get_symbol_table()`: Retrieves the symbol table - a data structure storing identifiers and their positions.

2. Tokenization:
   - Regular expressions are used to identify and extract tokens from the code.
   - Tokens are categorized based on their type (e.g., keywords, identifiers, operators).

3. Error Handling:
   - Lexical errors such as invalid tokens or unrecognized characters are identified and reported with line and column information.

Usage

Documentation

1. Input: Provide code files (`*.txt`) written in the custom programming language.

2. Execution:

   - Instantiate `MyScanner` with the input file path.

   - Call the `scan()` method to analyze the code and detect lexical errors.

3. Output:

   - The Program Internal Form (PIF) and Symbol Table files are generated with token information.