GitHub Link -Geanovu Medeea-Elena 933/1

Scanner Implementation

The Scanner class performs lexical analysis following these steps: it reads the tokens from a specified input file, categorizes them as constants, identifiers, operators, separators, reserved words, or methods, and generates a symbol table and PIF corresponding to the resulted symbols.

Attributes of the Scanner:

-separators: The separators read from Token.in.

-operators: The operators read from Token.in.

-reservedWords: The reservedWords read from Token.in

-symbolTable: An instance of SymbolTable

-pif: An instance of PIF

Token Classification:

-isWhitespace(String token): Checks if a token is a whitespace character.

-isValidConstant(String str): Validates if a string represents a valid constant (integer, double, float, string, or character).

- isValidIdentifier(String str): Validates if a string is a valid identifier according to the Purple language conventions (can start with underscore or a letter, but not with a number, then can be followed only by letters, numbers or more underscores).

-isSeparator(String token): Checks if a token is a defined separator.

-isOperator(String token): Checks if a token is a defined operator.

-isReservedWord(String token): Checks if a token is a reserved keyword.

-isMethod(String token): Checks if a token represents a method call.

Additional code checks for negation of variables and strings composed of multiple words.

-readCodificationTable(): Reads the separator, operator, and reserved word definitions from "token.in".

-insertInPIF(String token, String type): Inserts a token into the PIF with its corresponding type (Constant or Identifier).

-checkForStringConstant(String token, StringTokenizer tokenizer): Handles multi-part string literals.

-scan(String fileName): The input file is read line by line, tokenizing each line and classifying each token, inserting valid tokens into the symbol table and PIF, and handling errors for invalid tokens.