Abrudan Rebeca Rafaela 931

https://github.com/911-Abrudan-Rebeca/FLCD/tree/main/L2

Implement a scanner (lexical analyzer): Implement the scanning algorithm and use ST from lab 2 for the symbol table.

Operations for class Scanner:

- __init__(self, filepath) -initializes an istance of the 'Scanner' class
- readFile(self) - reads the content of the source code file line by line and returns it as a single string.
- def getProgramTokens(self) -tokenizing part, extracts individual tokens
- def scan(self) -lexical analysis, creates pif and checks for lexical errors
- def get_pif(self) -get pif (program internal form)
- def get_constantST(self) -get constants symbol table
- def get_identifiersST(self) -get identifiers symbol table

Variables:

- operators -list of operators
- separators -list of separators
- reservedWords -list of reserved keywods
- constantST -symbol table for constants
- identifierST -symbol table for identifiers
- pifOutput -list of the token-index pif
- filePath -the path of the source code file

The scanner tokenizes the source code files, categorizes the tokens, then builds a program internal form(pif).

The method getProgramTokens() is the tokenizing part. Firstly, it reads the source code from a given file, then it iterates through each character found. The tokens are temporarily added to the 'local_word' variable, if they are not operators/separators. If a separator/operator is found, the 'local_word' is added to the 'tokens' list, then the 'local_word' is reset to empty. Also, there is a case for quoted strings, which are considered a single token.

The method scan() is responsible for the lexical analysis, categorizing the tokens. Firstly, it gets the tokens from the previous method and checks the 'tokens' list not to be empty. Then, it iterates through each token. For the reserved words, operators, or separators, the value in 'pifOutput' is set to '-1'. For identifiers (names, names with numbers or _, regular expression) it assigns a number starting from 1 ('identifier_counter'). If the identifier already is in the symbol table, then it uses the index that was already assign to it. If not, it adds the identifier in the symbol table, with the incremented 'identifier_counter'. In the end it adds it to the 'pifOutput'. For the constants, it is the same as for the identifiers, but the constants can be integers, characters or strings (regular expression) and they have their own symbol table 'constantsST'. If the token is none of the above, it means it is a lexical error, which will be printed with the line of the code. If the 'lexical_error_exists' flag is not True, then it prints that the program is lexically correct.