CMPE 230 Systems Programming Homework 1

This project is about developing a translator that generates LLVM IR code that computes and prints arithmetic operations. There are two types of lines: assignment statements and expressions. In assignment statements, we are supposed to print the LLVM instructions that are necesssary for the calculation. In expressions, we are supposed to print the LLVM instructions that are necesssary for printing the value of the expression. In addition to them, there may be also some errors in both types of lines. Error codes are the followings:

1. Undefined variable (variables that are not allocated earlier)
2. Missing paranthesis (there are unmatched paranthesis)
3. Missing variable (not enough operands near operator)
4. Wrong named variable (identifiers that start with a digit or identifiers that are not alphanumeric)
5. Left hand side contains operators
6. No left or right hand side (left or right hand side is an empty string)
7. Assignment statements contain more than one assignment operators

For the solution, I decided to take on a simple algorithm. Steps are the followings:

- Check whether the line is an assignment statement or an expression.
- If it is an assignment statement
    - Split it into two and evaluate the right hand side.
    - Assign its value to left hand side.
- If it is an expression
    - Evaluate the expression.
    - Print its result.
- Evaluation
    - Get rid of all paranthesis by evaluating expressions inside them from innermost ones to outermost ones.
    - Compute all multiplication operations.
    - Compute all division operations.
    - Compute all subtraction operations.
    - Compute all addition operations.
- Computation
    - Extract the operands.
    - Get temporary variables if necesssary.
    - Evaluate the operation.
    - Write the temporary variable that contains the result in the place of operands and operator.

I have chose Java because its the language that I feel most comfortable and it has a wide range of functions for string manipulation.

I have chose to implement only a single class with static methods because the implementation is already easy and appropriate for procedural programming.

Fields and Their Functionalities

- TreeSet VARIABLES : Stores variables of LLVM.
- int lineCounter : Counter for input file lines.
- LinkedList list : Stores the strings to be written to the output file.
- int tempCounter : Counter for temporary variables of LLVM.

Methods and Their Functionalities

- void main(String[] args) throws FileNotFoundException
    - Read lines from input file, processes them and prints the LLVM code to output file.
    - parameter **args** path for input file
    - throws **FileNotFoundException** if input file is not found
- void errorMissingParanthesis(int index, String type)
    - Checks whether all parenthesis are matching.
    - parameter **index** return value from indexOf("(") or indexOf(")")
    - parameter **type** left or right parenthesis
- void errorMissingVariable(String expression, int index, String operator)
    - Checks whether there is two numbers or variables around the operators.
    - parameter **expression** string containing variables and operators
    - parameter **index** variable's supposed starting or ending point
    - parameter **operator** operator type
- String expression(String expression)
    - Processes all operations for all operators in a given string.
    - First removes all parenthesis then processes the operations according to the operator precedence.
    - All operations of a type are finished before the next operation type is executed.
    - parameter **expression** string containing variables and operators

    - returns processed string
- String findVariable(String expression, String temp)
    - Checks whether given string is a number or variable.
    - If it is a number then returns it.
    - If it is a variable first checks for errors then returns it as a temp variable.
    - parameter **expression** string consisting of number or temp variable
    - parameter **temp** temp variable of LLVM
    - returns number of temp variable of LLVM
- String operation(String expression, String operator, String type)
    - Processes all operations for a given operator in a given string.
    - First extracts the left and right operands then checks for errors then does the operation and changes the string for this operation then returns this string.
    - parameter **expression** string containing variables and operators
    - parameter **operator** operator type
    - parameter **type** LLVM code of operation
    - returns processed string