CS224 – Project 1: Jack Tokenizer

Background

The first step in writing a compiler for the Jack language is to chunk the characters in the file into a list of *tokens*. Each token is of a particular type like an identifier, number, string, or symbol.

Objective

You will complete the JackTokenizer by writing the `advance` method which determines the next token and token type in the input file. You should start with the state diagram for the tokens and turn this diagram into code.

Criteria for Success

The test files have been provided in Canvas along with the results that you should obtain. When you compare your resulting xml files with the test files, using TextComparer they should be identical.

Resources

The relevant reading for this project is Chapter 10 in the nand2tetris text. I have also supplied a zip file containing starter files.

Tools

The TextComparer is available on phoenix. Run the TextComparer in the terminal supplying the two files that are being compared as arguments.

Hints

Be sure that you implement the state diagram in the way we did in the class activity rather than trying to use nested if statements. Nested if statements will lead to suffering.

The `advance` method assigns values to the fields `tokenType`, `keyWord`, `symbol` and `identifier` as appropriate. For example, if the token is an integer constant you would set the field `tokenType` to `TokenType.INT_CONST` and set the field `identifier` to the string of digits for that integer constant.

The `TokenType` class is provided for you and an incomplete `State` class has also been provided.

Submission and Assessment

Submit in Canvas your entire JackTokenizer project as a single ZIP archive. Your project will be graded using the rubric provided in the Canvas submission.