Make build removes the output generated files and rebuilds the lexer. From here, ./lexer < 29_A2.nc will run it with the test file. Make run would also do the same.

1. Introduction

nanoC Lexer README

This document describes the lexical grammar for the nanoC programming language and how the

provided lexer recognizes and tokenizes the source code written in this language.

- Token Recognition:
Tokens are recognized based on regular expressions defined in the Flex file.
-Ignoring Whitespace: Whitespace characters (space, tab, newline) are ignored by the lexer.
-Error Handling: Invalid input is handled by printing an error message and continuing tokenization.

2. Lexical Elements

The primary lexical elements (tokens) that the lexer identifies in the source code are:

- keyword

- identifier

- constant

- string-literal

- punctuator

3. Keywords

The keywords for nanoC are: char, else, for, if, int, return, and void.

4. Identifiers

Identifiers are sequences of letters and digits, but they must begin with a letter. Both lowercase (a to

z) and uppercase (A to Z) letters are valid starting characters for identifiers.

5. Constants

There are two types of constants: integer-constant and character-constant. Integer constants are whole numbers, while character constants are characters enclosed in single quotes.

6. String Literals

String literals are sequences of characters enclosed in double quotes. The string is terminated by a null character ('\0').

7. Punctuators

Punctuators are symbols used in the syntax of the language, including common symbols and compound operators like <=, >=, etc.

8. Comments

The lexer handles both single-line and multi-line comments:
- Single-line comments: Comments that start with `//` and continue until the end of the line.
-Multi-line comments: Comments enclosed within `/*` and `*/`.