Monsoon 2023 CS1319 A2

Bhumika Mittal

Introduction

The lexical analyzer scans the stream of characters of the source program and groups characters into lexemes (the smallest meaningful unit of a program). For each lexeme, the lexer outputs a token that encapsulates the token name and a pointer to the corresponding entry of the lexeme in the symbol table.

Design Choices

The lexer classifies the given nanoC code into the five following tokens: keyword, identifier, constant, string-literal, punctuator. The tokens, with syntax error, generates an INVALID TOKEN and terminates the lexer without analysing the program any further. The comments (both single line and multiple line) and whitespaces are ignored. The definitations of the tokens are as per the specifactions mentioned in the given assignment document.

It is an interactive flex as it repeatedly calls yylex(). Every call returns one token (after taking the actions for the rule matched) that is consumed by the parser and yylex() is again called for the next token. This lets parser and lexer work hand-in-hand and also eases information interchange between the two.

keyword

If the lexer detects any word among char, else, for, if, int, return, void, then it classifies it as a keyword.

sign, digit & non-digit

If any alphabet, (smallcase or capital) or _ is detected, it is considered as non-digit. Similarly, any number between 0-9 is a digit. Any digit between 1-9 is a nonzero digit. Similarly, any combination of a non-digit, and digit is classified as nondigit digit. + or - are classified as sign.

This will be useful later to define identifier, constant, string-literal.

escape sequence

These are the combination of characters that has a meaning other than the literal characters contained therein. They are dealt with separately.

identifier

They are basically nondigits followed by 0 or more nondigit digit. This pattern is designed to match sequences that start with a non-digit character, followed by zero or more occurrences nondigit digit.

interger constant

sign is optional,followed by some nonzero digit and then 0 or more digit or just 0.

c_char

This matches any character except the single quote, a backslash, or a newline character or the caret. Also, escape sequence may also be matched.

c_char sequence.

One or more occurrence of c_char is classified as c_char sequence.

character constant

This matches something that starts and ends with single quotes and contains one or more characters (excluding single quotes, backslashes, and newline characters) in between.

constants

It matches either integer constant or a character constant

s_char

This matches any character except the double quote, a backslash, or a newline character or the caret. Also, escape sequence may also be matched.

s_char sequence.

One or more occurrence of s_char is classified as s_char sequence.

string literal

This matches something that starts and ends with double quotes and contains one or more characters (excluding double quotes, backslashes, and newline characters) in between.

punctuator

It matches any one of the character defined as punctator according to the assignment document.

whitespace

It matches any space, tab, or newline character.

multi line & single line comment

It is used to recognize the start of a multi-line (or single line) comment and transition to this state. In this state, the lexer would continue reading characters until it encounters the end of the multi-line comment (or single line), and then it would transition back to the default state.

invalid token

It matches everything. We will put this rule towards the end so that if one of the other rule is satisfied then this is satisfied.