Manual Scanner vs JFlex Scanner Comparison

Abdul Mohaimin (23i-0652) | Shazil Rehman (23i-0095)

Introduction

In this document we present a comparative analysis of two lexical analyzer implementations for the CC-Lang programming language: a manually implemented DFA-based scanner and an automatically generated JFlex scanner. Both scanners were tested on identical input files to evaluate their correctness, consistency, and performance. The comparison focuses on token recognition accuracy, error handling capabilities, and execution efficiency.

Output Comparison

Test File	Manual Scanner Output	JFlex Scanner Output	Differences
test1.lang	Total Tokens: 59 Lines Processed: 16 Comments Removed: 2 Token counts by type: OPERATOR: 22 KEYWORD: 7 BOOLEAN_LITERAL: 2 INTEGER_LITERAL: 3 IDENTIFIER: 6 PUNCTUATOR: 17 FLOAT_LITERAL: 1 STRING_LITERAL: 1	Total Tokens: 58 Lines Processed: 17 Comments Removed: 0 Token counts by type: IDENTIFIER: 6 OPERATOR: 21 INTEGER_LITERAL: 3 PUNCTUATOR: 17 STRING_LITERAL: 1 KEYWORD: 7 FLOAT_LITERAL: 1 BOOLEAN_LITERAL: 2	• **Cause**: The manual scanner counts comments as tokens before filtering them out, while JFlex silently skips comments without tokenizing them. • **Impact**: No functional difference as both produce correct output tokens.
test2.lang	Total Tokens: 49 Lines Processed: 8 Comments Removed: 0 Token counts by type: OPERATOR: 13 KEYWORD: 7 INTEGER_LITERAL: 5 IDENTIFIER: 12	Total Tokens: 49 Lines Processed: 9 Comments Removed: 0 Token counts by type: IDENTIFIER: 12 OPERATOR: 13 INTEGER_LITERAL: 5 PUNCTUATOR: 12	

	PUNCTUATOR: 12	KEYWORD: 7	
test3.lang	Total Tokens: 18 Lines Processed: 6 Comments Removed: 0 Token counts by type: OPERATOR: 2 KEYWORD: 6 CHARACTER_LITERAL: 2 IDENTIFIER: 2 PUNCTUATOR: 4 STRING_LITERAL: 2	Total Tokens: 18 Lines Processed: 7 Comments Removed: 0 Token counts by type: IDENTIFIER: 2 OPERATOR: 2 CHARACTER_LITERAL: 2 PUNCTUATOR: 4 STRING_LITERAL: 2 KEYWORD: 6	
test4.lang	Total Tokens: 25 Lines Processed: 10 Comments Removed: 6 Token counts by type: OPERATOR: 5 KEYWORD: 7 INTEGER_LITERAL: 3 IDENTIFIER: 3 PUNCTUATOR: 5 FLOAT_LITERAL: 2	Total Tokens: 34 Lines Processed: 11 Comments Removed: 0 Token counts by type: IDENTIFIER: 12 OPERATOR: 5 INTEGER_LITERAL: 4 PUNCTUATOR: 5 KEYWORD: 7 FLOAT_LITERAL: 1	• **Cause**: Different error recovery strategies. The manual scanner skips malformed tokens entirely, while JFlex attempts to tokenize parts of invalid input. • **Impact**: JFlex provides more detailed error information by tokenizing recoverable portions of erroneous input.
test5.lang	Total Tokens: 5 Lines Processed: 11 Comments Removed: 4 Token counts by type: KEYWORD: 3 PUNCTUATOR: 1 STRING_LITERAL: 1	Total Tokens: 7 Lines Processed: 12 Comments Removed: 0 Token counts by type: OPERATOR: 1 INTEGER_LITERAL: 1 PUNCTUATOR: 1 STRING_LITERAL: 1 KEYWORD: 3	• **Cause**: Similar to test4.lang, JFlex's error recovery tokenizes more elements from partially valid input.

Performance Analysis

Execution times for test1.lang gave the following results:

- Manual Scanner Time: **117.2249 ms**
- JFlex Scanner Time: **101.8088 ms**

The JFlex scanner demonstrated superior performance with an execution time of 101.8 ms compared to the manual scanner's 117.2 ms (approximately 13% faster). This efficiency gain is expected, as JFlex generates optimized state transition tables and uses efficient pattern matching algorithms.