Course Name:	Operating Systems and Compilers	Semester:	VI
Date of Performance:	11 / 04 / 2025	Batch No.:	B - 2
Faculty Name:	Prof. Nilesh Lakade	Roll No.:	16014022050
Faculty Sign & Date:		Grade/Marks:	___ / 25

Experiment No.: 9

Title: To learn about lexical analyzer in compiler design

Aim and Objective of the Experiment:

To learn about lexical analyzer and different tokens such as identifiers, keywords, operators, punctuation symbols, and constants in compiler design.

COs to be achieved:

CO5: Understand compiler construction tools and describes the functionality of each stage of compilation process.

Theory:

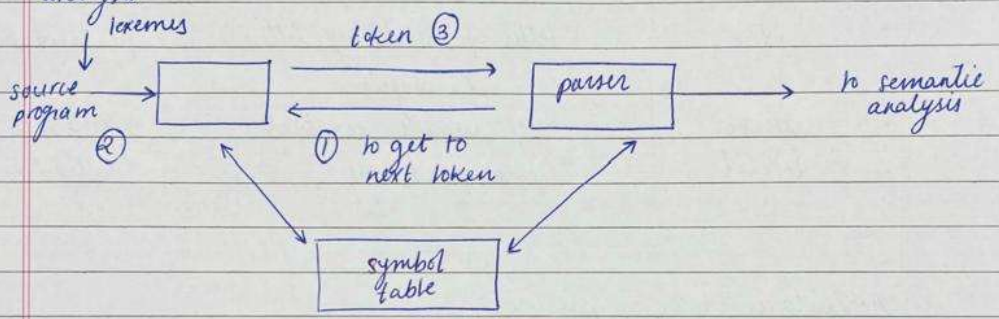
Ketaki M
16014022050

classmate
Date 18/04/25
Page

OSC experiment 9: Lexical Analyzer in Compiler Design

For a compiler, the first phase is to do lexical analysis. The main task of a lexical analyzer is to read the input characters of the source program, group them into lexemes and produce as output a sequence of tokens for each lexeme in the source program.

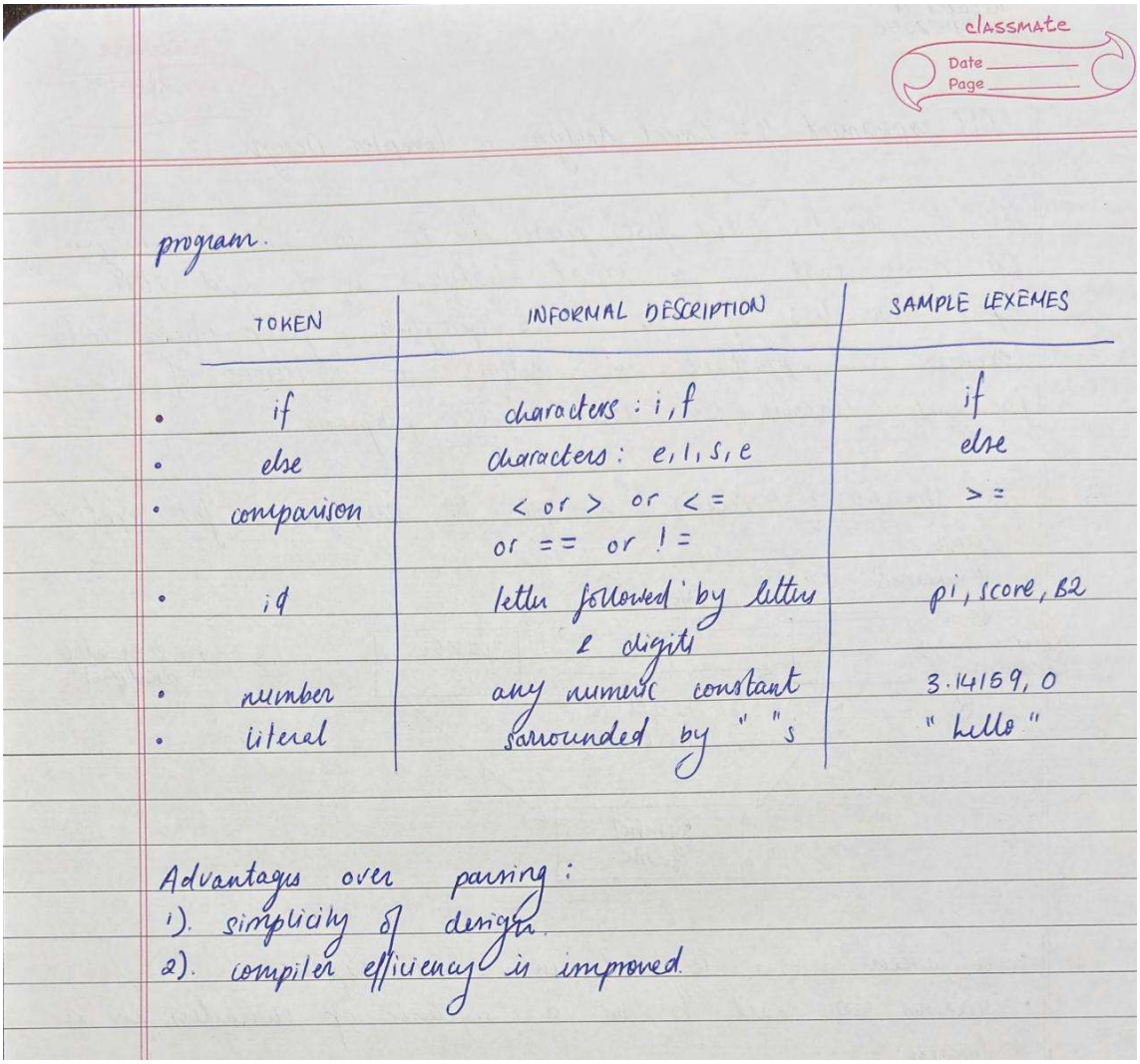
This stream of tokens is sent to the parser for syntax analysis.



- a token is a keyword in a set sentence.
- lexemes are said to be a sequence of characters in a token.
- a pattern explains what can be a token, defined by regular expressions.
- an identifier is a token used to identify elements in a program.

When the lexical analyzer discovers a lexeme consisting of an identifier, it adds the lexeme into the symbol table if not present.

The lexical analyzer skips out comments & blankspaces (whitespaces newline, tab, etc.). It also correlates error messages generated by the compiler with the source



program.

TOKEN	INFORMAL DESCRIPTION	SAMPLE LEXEMES
• if	characters : i, f	if
• else	characters : e, l, s, e	else
• comparison	< or > or <= or == or !=	>=
• id	letter followed by letters & digits	pi, score, D2
• number	any numeric constant	3.14159, 0
• literal	surrounded by " "s	"hello"

Advantages over parsing :
1). simplicity of design.
2). compiler efficiency is improved.

References:

- Compilers: Principles, Techniques, and Tools is a computer science textbook by Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D

Conclusion:

Thus, understanding lexical analysis helps in recognizing how source code is broken into tokens for easier parsing. It also shows how compilers detect errors and links them to specific locations in the code, making debugging more efficient.

Signature of faculty in-charge with Date: