Report on lexical analysis of my language

Name	Md Abu Saeed
Roll	**1907057**
Dept	**CSE**
Group	**A2**
Course-No	**CSE-3212**
Course-Title	**Compiler Design Laboratory**

Objective:

1. To learn lexical analysis implementation on our custom input.
2. To identify various language constructs.
3. To learn about implementation of basic language concepts.
4. To learn implementation of bison.
5. To create a basic custom compiler.

Introduction:

Lexical analysis is a crucial first step in compiling code. It involves breaking down the source code into tokens, which represent meaningful code elements. This process helps convert human-readable code into a format that computers can understand and process. Whereas bison is for matching token according to our rules. We can define our custom rules that follows the structure of our custom language. Tokens passed from the flex is used for matching with our custom defined rules.

Discussion:

Source code is scanned and divided into distinct tokens. Source code is divided into multiple virtual sub-section(like MAIN_SEC, IF_SEC, COMMENT_SECTION, LOOP_SEC etc) to have good control over the user input. Then I have used regex to match the input string based on the section where control is now located. Then I have passed token according to my language structure to bison file for further operations. Here I have covered token identification and categorization of **variables**, **Data types**, **Arithmetic and Conditional Operators**, **Control Structures**, **Comments**, **Loop, Function** etc

Conclusion:

Lexical analysis is a foundational process in compiler design. It serves as the initial step in translating human-readable source code into a format suitable for computer processing. By identifying and categorizing tokens, lexical analysis helps in execution of code. Whereas bison uses the tokens from the flex and matches it with our rules to ensure that it follows the correct order and structure of our language. After matching a rule, further operation can be performed on it to process result to be used later. It plays a vital role in the compilation process and enables the development of efficient and functional language.