DAY -36, DAILY REPORT, 06 -01 -2022 (Thursday)

Today, I had an experience. When I woke up in the morning, it gave me a positive vibe, and the cab came to the apartment and I got into it and I went to the office safely. I got selected in surfboard payments india private limited as a position of security engineer. After that I attended the session in the morning with askin and my intern batch mates, he talked about how to compile and run the program using javascript. First of all I want to learn about a compiler that is a software that converts the source code to the object code. In other words, we can say that it converts the high level language to machine/binary language. Moreover, it is necessary to perform this step to make the program executable. This is because the computer understands only binary language. Some compilers convert the high-level language to an assembly language as an intermediate step. Whereas some others convert it directly to machine code. This process of converting the source code into machine code is called compilation. We can analyze a source code in three main steps. Moreover, these steps are further divided into different phases. The three steps are: Linear Analysis - Here, it reads the character of the code from left to right. The characters having a collective meaning are formed. We call these groups tokens. Hierarchical Analysis - According to collective meaning, we divide the tokens hierarchically in a nested manner. Semantic Analysis - In this step, we

check if the components of the source code are appropriate in meaning. The compilation process takes place in several phases. Moreover, for each step, the output of one step acts as the input for the next step. The phases/structure of the compilation process is as follows: Lexical Analyzer - It takes the high-level language source code as the input. It scans the characters of source code from left to right. Hence, the name scanner also. It groups the characters into lexemes. Lexemes are a group of characters which have some meaning. Each lexeme corresponds to form a token. It removes white spaces and comments. It checks and removes the lexical errors.Syntax Analyzer - 'Parser' is the other name for the syntax analyzer. The output of the lexical analyzer is its input. It checks for syntax errors in the source code. It does this by constructing a parse tree of all the tokens. For the syntax to be correct, the parse tree should be according to the rules of source code grammar. The grammar for such codes is context-free grammar.Semantic Analyzer - It verifies the parse tree of the syntax analyzer. It checks the validity of the code in terms of programming language. Like, compatibility of data types, declaration, and initialization of variables, etc. It also produces a verified parse tree. Furthermore, we also call this tree an annotated parse tree. It also performs flow checking, type checking, etc. Intermediate Code Generator (ICG) - It generates an intermediate code. This code is neither in high-level language nor in machine language. It is in an intermediate form. It is converted to machine language but the last two phases are platform dependent.The intermediate code is the same for all the

compilers. Further, we generate the machine code according to the platform. An example of an intermediate code is three address codes. Code Optimizer - It optimizes the intermediate code. Its function is to convert the code so that it executes faster using fewer resources (CPU, memory).It removes any useless lines of code and rearranges the code.The meaning of the source code remains the same. Target Code Generator - Finally, it converts the optimized intermediate code into the machine code.This is the final stage of the compilation.The machine code which is produced is relocated. Thank you, that's all for today.