Aim: Construct LL(1) parser and check for the acceptance of the string using JFLAP (Java Formal Languages and Automata Package) tool.

Objectives: • To construct a parser and verify the acceptance of a given string using the JFLAP tool.

• To analyze the working of formal languages and automata in parsing. To demonstrate the correctness of the designed parser.

Introduction: Parsing is a fundamental concept in compiler design and formal language processing. A parser analyzes input sequences based on a given grammar to determine their syntactical structure. In this project, we will construct a parser for the given grammar and check string acceptance using JFLAP, a widely used tool for automata and formal language processing. The parser is one of the phases of the compiler which takes a token of string as input and converts it into the corresponding Intermediate Representation (IR) with the help of an existing grammar.

The parser is also known as Syntax Analyzer. Conditions for an LL(1) Grammar To construct a working LL(1) parsing table, a grammar must satisfy these conditions:

• No Left Recursion: Avoid recursive definitions like A -> A + b.

• Unambiguous Grammar: Ensure each string can be derived in only one way.

• Left Factoring: Make the grammar deterministic, so the parser can proceed without guessing. Algorithm to Construct LL(1) Parsing Table

Step 1: First check all the essential conditions mentioned above and go to step 2.

Step 2: Calculate First() and Follow() for all non-terminals. 1. First() : If there is a variable, and from that variable, if we try to drive all the strings then the beginning Terminal Symbol is called the First. 2. Follow() : What is the Terminal Symbol which follows a variable in the process of derivation.

Step 3: For each production A –> α. (A tends to alpha) 1. Find First(α) and for each terminal in First(α), make entry A –> α in the table. 2. If First(α) contains ε (epsilon) as terminal, then find the Follow(A) and for each terminal in Follow(A), make entry A –> ε in the table. 3.

If the First(α) contains ε and Follow(A) contains $ as terminal, then make entry A –> ε in the table for the $. Problem / Goal / Proposed System: The problem at hand is to design and implement a parser for the given context-free grammar (CFG) defined as: S -> aABb A -> c/€ B -> d/€ The primary goal of this project