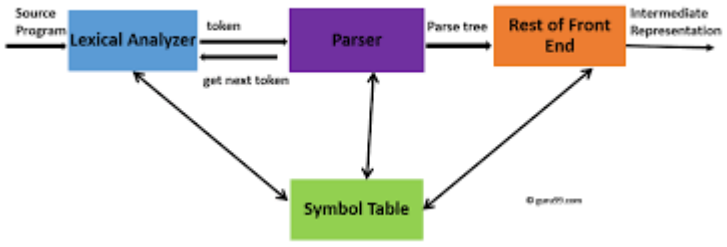
Minor Project- Report
June-2022

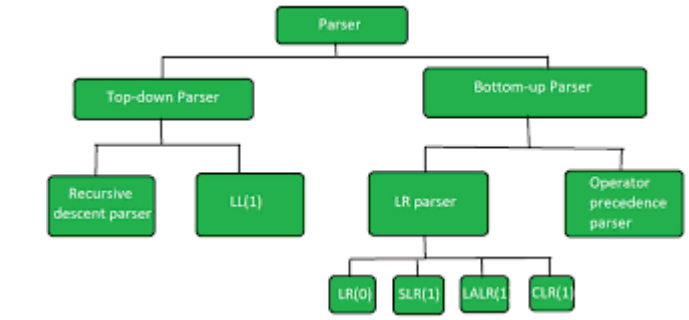
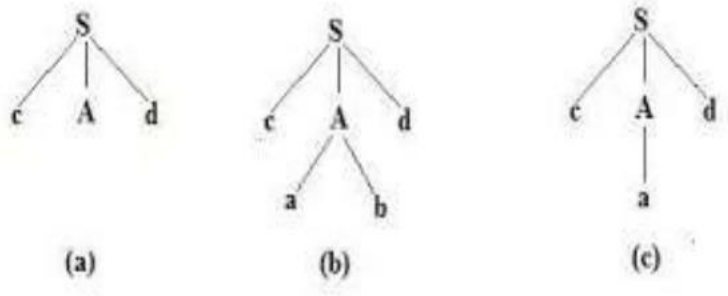
Course Faculty: Dr. Vindhya P Malagi, Prof. Muquitha Almas, Prof. Shravya AR
Course Name: System Software and Operating System Laboratory
Course Code: 19CS6DLSSL
Semester: 6th
Date: 21-06-2022

TITLE OF THE PROJECT	**Implementation of a Recursive Descent Parser**			
STUDENT NAME	Kanishk Bhaskar	Karan V	Keshav Jhawar	Kiran N Deshmukh
USN	1DS19CS067	1DS19CS068	1DS19CS069	1DS19CS070
GUIDE	Dr. Vindhya P Malagi			

PROJECT ABSTRACT:	We have implemented the Recursive Descent Parser as part of our mini project in C Programming Language.
	
	The algorithm for Recursive Descent Parser is as follows:
	➢ Consider the grammar:
	S -> cAd
	A -> ab|a
	➢ Let the input string be 'w' = cad.
	➢ Procedure S is written as follows:
	procedure S ()
	begin
	if input symbol = 'c' then
	ADVANCE ();
	if A () then
	if input symbol = 'd' then
	begin ADVANCE ();
	return true;
	end
	end;
	return false;
	end			

	➢ Procedure A is written as follows:
	Procedure A ()
	begin
	isave := input-pointer;
	if input symbol = 'a' then
	begin
	ADVANCE ();
	if input symbol = 'b' then
	begin ADVANCE (); return true;
	end
	end
	input-pointer := isave;
	/* failure to find ab */
	if input symbol = 'a' then
	Begin ADVANCE (); return true;
	end
	else return false;
	end

INTRODUCTION	Parsing is the process to determine whether the start symbol of a specific grammar can derive the program. ➢ If successful, the program is a valid program. ➢ If failed, the program is invalid. There are two approaches to parsing in general. ➢ Expanding from the start symbol to the whole program (Top Down). Ex. Recursive Descent Parser, LL Parser or Predictive Parser ➢ Reduction from the whole program to the start symbol (Bottom Up). Ex. LR (SLR, Canonical LR, LALR) Parser Recursive Descent Parsing is a Top-Down method of syntax analysis in which a set of recursive procedures to process the input is executed. A procedure is associated with each non-terminal of a grammar. Top-Down parsing can be viewed as an attempt to find a leftmost derivation for an input string. It attempts to construct a parse tree for the input starting from the root and creating the nodes of the parse tree in preorder. Recursive Descent Parsing involves backtracking.
DESIGN	

	 
PLATFORM USED	C Programming Language, VS Code
PROJECT SOURCE CODE LINK	https://github.com/kirandeshmukh6421/recursive-descent-parser
CONCLUSION /FUTURE ENHANCEMENT	The parser could be improvised to use a non-recursive descent parsing algorithm which will not require any kind of backtracking to improve parse speeds and efficiency. These types of parsers are also called Predictive Parsers.

SCREENSHOTS