Compiler design

project proposal –

operator precedence parser

5th-April-2022

**GROUP MEMBERS: CT-18057 NOOR SHAUKAT & CT-18054 MUHAMMAD ATHAR SAYEED**

# Overview

## Project Description

|  |  |
| --- | --- |
|  | *An operator-precedence parser is a simple shift-reduce parser that is capable of parsing a subset of LR (1) grammars or in simple words an operator precedence parser is a*[*bottom-up parser*](https://en.wikipedia.org/wiki/Bottom-up_parsing)*that interprets an*[*operator-precedence grammar*](https://en.wikipedia.org/wiki/Operator-precedence_grammar)*.*  *For example, most*[*calculators*](https://en.wikipedia.org/wiki/Calculator)*use operator precedence parsers to convert from the human-readable*[*infix notation*](https://en.wikipedia.org/wiki/Infix_notation)*relying on*[*order of operations*](https://en.wikipedia.org/wiki/Order_of_operations)*to a format that is optimized. More precisely, the operator-precedence parser can parse all LR (1) grammars where two consecutive nonterminal and epsilon never appear in the right-hand side of any rule.*  *Operator precedence grammar is kinds of shift reduce parsing method. It is applied to a small class of operator grammars.*  *A grammar is said to be operator precedence grammar if it has two properties:*   * *No R.H.S. of any production has a∈.* * *No two non-terminals are adjacent.*   *Operator precedence can only be established between the terminals of the grammar. It ignores the non-terminal.* |

## Project Scope

|  |  |
| --- | --- |
|  | *Operator-precedence parsers are not used often in practice - however, they do have some properties that make them useful within a larger design.*  *First, they are simple enough to write by hand, which is not generally the case with more sophisticated right shift-reduce parsers.*  *Second, they can be written to consult an operator table at*[*run time*](https://en.wikipedia.org/wiki/Run_time_(program_lifecycle_phase))*, which makes them suitable for languages that can add to or change their operators while parsing.* |

## Deliverables

|  |  |
| --- | --- |
|  | Implementation includes, (Acceptance and Rejection of the Input) |

The parser will identify the following:

* If the input keeps in step with the grammar.
* If two operators are given together in an input.
* If the input has an operator at the very end.