Experiment 5

Aim: Write C programs to implement the following.

A. Find the first and follow of all non-terminals in any given grammar.

B. Construct a recursive descent parser for the grammar.
$E \rightarrow E + T | T$
$T \rightarrow T * F | F$
$F \rightarrow (E) | id$

A. Find the first and follow of all non-terminals in any given grammar

The program must:

- Take a set of production rules as input.
- Input the no of productions followed by each production.
- Calculate the first and follow sets for each non-terminal in the production rules and display them..

First Set

First(α) is a set of terminal symbols that begin in strings derived from α.

Follow Set

Follow(α) is a set of terminal symbols that appear immediately to the right of α.

B. Recursive descent parser

Construct a recursive descent parser for the grammar.

$E \rightarrow E + T | T$

$T \rightarrow T * F | F$

$F \rightarrow (E) | id$

Remember

It can be clearly seen that, the grammar has left-recursion, which needs to be eliminated before using it to build a recursive descent parser.

On eliminating left recursion, we have

$E \rightarrow TE'$

$E \rightarrow +TE' | \epsilon$

$T \rightarrow FT'$

$T' \rightarrow *FT' | \epsilon$

$F \rightarrow (E) | id$

Implementation

- RDP can be easily implemented by using a recursive procedure for each of the non-terminals.
- On each production, the corresponding recursive procedures are called.
- The input is matched when a terminal is encountered in a production.
- We can ignore the ϵ, since it represents the empty string.