

Compiler Assignment 2

Name: Ameer Ayman Ahmed

ID: 135058

After first stage of designing the compiler which is lexical analyzer. Which take all the code and split it into tokens (tokenization). We go through the syntax analyzer, it depends on the lexical and it take the tokens, read it one by one and go through the grammar of the compiler to check if the code is follow the grammar or not. Before following the grammar to check the correctness of the code, we need to make sure that this grammar is LL (1) and all production rules is clear. We clean the grammar first and check that it does not contain unnecessary productions that cannot be used.

The grammar transformations have two types:

1. Removal of left recursion, by introducing new non-terminal.
2. Factorization, by factoring out common symbols which can be lead to ambiguity.

These are the rules that have left recursion and the transformation of them.

1. Identifier\_list id W

W  , id W

W $

1. Declarations  D

D  var identifier\_list : type ; D

D  $

1. subprogram\_declarations  C

C subprogram\_declaration ; C

C $

1. Parameter\_list  identifier\_list : type B

B ; identifier\_list : type B

B $

1. Statement\_list  statement A

A ; statement A

A $

1. Expression\_list  expression Z

Z  , expression Z

Z  $

1. Simple\_expression  term Y

Y addop term Y

Y $

1. Term factor X

X mulop factor X

X $

There are three rules that have factorization, and this is the transformation of them

1. Variable  id variable2

Variable2  [expression]

Variable2 $

1. Expression  simple\_expression expression2

Expression2  relop simple\_expression

Expression2  $

1. Procedure\_statement  id Procedure\_statement2

Procedure\_statement2  (expression)

Procedure\_statement2  $

The program takes token by token from the file which is the output of lexical and go through the grammar to reach all the terminal. The code run by recursion, it goes from one non-terminal to another until reach the terminals. If there are any errors found by the parser while checking, it shows the line number, the reason and what should have expected.