

**Cairo University**

**Faculty of Engineering**

**Computer Department – Forth Year**

**Compilers Project**

*Submitted to:*

**Eng. Yousef Ghattas**

*Submitted by:*

**Omar Adel Moutaz Samir**

**(SEC : 2 - BN : 3) (SEC : 2 - BN : 19)**

**Ramadan Ahmed Tarek Mohamed**

**(SEC : 1 - BN : 23) (SEC : 1 - BN : 28)**

Project Overview

A programming language that supports basic data types (int) and supports also assignment statements and mathematical and logical expressions, it also provides control statements and conditional statements.

Tools and Technologies Used

* Python
* Windows (Operating System)
* C# WinForms ( Graphical User Interface GUI)

List of Tokens

|  |  |
| --- | --- |
| Token | Description |
| INT | Integer data type |
| CONST | Constant data type |
| IF | If keyword (conditional) |
| ELSE | Else keyword |
| FOR | For loop |
| WHILE | While loop |
| DO | Do keyword |
| SWITCH | Switch keyword |
| CASE | Case (option in a switch statement) |
| BREAK | Break keyword |
| DEFAULT | Default (default case in switch statement) |
| TRUE | True keyword ( not equal to zero) |
| FALSE | False keyword (equal to zero) |
| AND | Logical and |
| OR | Logical or |
| NO­­T | ! |
| EQUAL | == |
| NOT\_EQUAL | != |
| LESS\_ EQUAL | <= |
| GREATER\_ EQUAL | >= |
| LESS | < |
| GREATER | > |
| COLON | : |
| COMMA | , |
| ASSIGN | = |
| SEMI\_COLON | ; |
| L\_PAREN | ( |
| R\_PAREN | ) |
| L\_BRACE | { |
| R\_BRACE | } |
| PLUS | + |
| MINUS | - |
| MULTIPLY | \* |
| DIVIDE | / |

Production Rules

* **type\_spec**
  + INTEGER\_TYPE
* **Program**
  + statement\_list
* **statement\_list**
  + statement
  + statement statement\_list
* **statement**
  + BREAK SEMI\_COLON
  + | selection\_statement
  + | iteration\_statement
  + | assignment\_statement SEMI\_COLON
  + | declaration\_statement SEMI\_COLON
  + | const\_statement SEMI\_COLON
  + | SEMI\_COLON
* **init\_for**
  + assignment\_statement
  + | declaration\_statement
* **compound\_statement**
  + L\_BRACE statement\_list R\_BRACE
  + | L\_BRACE R\_BRACE
* **assignment\_statement**
  + variable ASSIGN expr
* **selection\_statement**
  + IF L\_PAREN boolean\_expression R\_PAREN compound\_statement
  + | IF L\_PAREN boolean\_expression R\_PAREN compound\_statement ELSE compound\_statement
  + | SWITCH L\_PAREN variable R\_PAREN L\_BRACE (CASE (INTEGER\_VALUE | MINUS INTEGER\_VALUE) COLON compound\_statement)+ (DEFAULT COLON compound\_statement) R\_BRACE
* **iteration\_statement**
  + WHILE L\_PAREN boolean\_expression R\_PAREN compound\_statement
  + | DO compound\_statement WHILE L\_PAREN boolean\_expression R\_PAREN SEMI\_COLON
  + | FOR L\_PAREN init\_for SEMI\_COLON boolean\_expression SEMI\_COLON assignment\_statement R\_PAREN compound\_statement
* **variable\_declaration**
  + TYPE variable ASSIGN expr
  + TYPE variable
* **const\_declaration**
  + CONSTANT TYPE variable ASSIGN expr
* **expr**
  + term ((PLUS | MINUS) term)\*
* **term**
  + factor ((MULTIPLY | DIVIDE) factor)\*
* **factor**
  + PLUS factor
  + | MINUS factor
  + | INTEGER\_VALUE
  + | LPAREN expr RPAREN
  + | variable
* **boolean\_expression**
  + boolean\_term (relation boolean\_term)\*
* **boolean\_term**
  + NOT boolean\_term
  + | TRUE
  + | FALSE
  + | L\_PAREN boolean\_expression R\_PAREN

List of Quadruples

|  |  |
| --- | --- |
| Quadruple | Description |
| (=, a , ,b) | b=a |
| (equal, a,b,Rx) | If (a ==b) Rx = true and false otherwise |
| (jfalse, Lx, Rx, ) | Jump to Lx if Rx is false |
| (jtrue, Lx, Rx, ) | Jump to Lx if Rx is true |
| (uminus, a, ,Rx) | Rx = -a |
| (+, a, b, Rx) | Rx = a + b |
| (-, a, b, Rx) | Rx = a - b |
| (\*, a, b, Rx) | Rx = a \* b |
| (/, a, b, Rx) | Rx = a / b |
| (AND, a, b, Rx) | Rx = a AND b |
| (OR, a, b, Rx) | Rx = a OR b |
| (Greater, a, b, Rx) | Rx = true if a > b and false otherwise |
| (LESS, a, b, Rx) | Rx = true if a < b and false otherwise |
| (jmp, Lx, , ) | Jump to Lx |