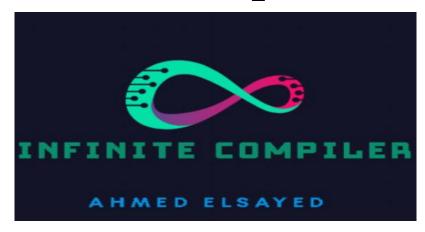
# الاسم: أحمد السيد السيد محمد جمعه Cs => 806326326



## Infinite Compiler...<>



- ⇒ Creating Parse Table.
- **⇒** Language Grammar to Create Parse Table?

```
program -> statement_list
statement_list -> statement ';' | statement_list ';'
statement -> assignment | print | if_statement | while_loop
assignment -> identifier '=' expression
print -> 'print' expression ';'
if_statement -> 'if' expression '{' statement_list '}'
while_loop -> 'while' expression '{' statement_list '}'
expression -> term | number op number
term -> number | identifier | string | '(' expression ')'
number -> digit
identifier -> '[a-zA-Z_][a-zA-ZO-9_]*'
op -> '+' | '-' | '*' | '/'
string -> '[a-zA-Z_][a-zA-ZO-9_]*'
digit -> '[0-9]+'
```

⇒ Terminals and Non-Terminals:

```
# Terminals
terminals = {
     ';', '=', 'print', 'if', '{', '}', 'while', '+', '-', '/', '(', ')',
     '[a-zA-Z_][a-zA-ZO-9_]*', '[0-9]+'
}

# Non-terminals
non_terminals = {
     'program', 'statement_list', 'statement', 'assignment', 'print',
     'if_statement', 'while_loop', 'expression', 'term'
}
```

**⇒** Constructing Parse Table using First and Follow Set:

#### ⇒ First Set:

```
First(program) = {identifier, print, if, while}
First(statement_list) = {identifier, print, if, while}
First(statement) = {identifier, print, if, while}
First(assignment) = {identifier}
First(print) = {'print'}
First(if_statement) = {'if'}
First(while_loop) = {'while'}
First(expression) = {number, identifier, string, '(')}
First(term) = {number, identifier, string, '(')}
First(number) = {digit}
First(identifier) = {'[a-zA-Z_]'}
First(op) = {'+', '-', '*', '/'}
First(string) = {'[a-zA-Z_]'}
First(digit) = {'[0-9]'}
```

#### **Follow Set:**

```
Follow(program) = $
Follow(statement_list) = Follow(program) = $
Follow(statement) = {';', '}', Follow(statement_list)}
Follow(assignment) = {';', '}', Follow(statement_list)}
Follow(print) = {';', '}', Follow(statement_list)}
Follow(if_statement) = {Follow(statement_list), '}'}
Follow(while_loop) = {Follow(statement_list), '}'}
Follow(expression) = {')', Follow(term)}
Follow(term) = {op, ')', Follow(expression)}
Follow(identifier) = {op, ')', Follow(term)}
Follow(op) = {number, identifier, string, '(')}
Follow(string) = {op, ')', Follow(term)}
Follow(digit) = {op, ')', Follow(term)}
```

### **Code Output**

```
VARIABLE NAME
                 OBJECT CODE ADDRESS
                                            DATA TYPE
                                                               NO OF DIMENSIONS
                                                                                     LINE OF DECLARATION
  REFRENCE LINE
PARSE TABLE:
Terminal
                                   identifier
                                                                 print ...
Non-terminal
statement_list statement; statement_list statement; statement_list ...
statement
                                   assignment print expression; ...
                 identifier = expression ;
assignment
expression
                                         term
                                   identifier
term
identifier identifier (expression) op term
string
                                   identifier
print
                                                     print expression ;
if statement
while_loop
number
digit
                                                                        ... * term / term
ор
```