April 7, 2001 **ICS 313-02 (002)** Dr. Mamdouh Najjar

Fundamentals of Programming Languages

Programming Assignment # 3
Interpreter for a First Language PP (*FirLan* ++)

(30 Points)

Due April 28, 2001

I. **Given *FirLan*++ Syntax described in BNF rules:**

<program> → <prog_name> PROG_START; <blocks> PROG_END;
<prog_name> → <ident>
<blocks> → <block> | <block> <blocks>
< block> → <datadef> <stmtblock>
<datadef> → DEFINE <datatypes> END;
<stmtblock> → BLOCK_BEGIN; <stmts> BLOCK_END;
<stmts> → <stmt> | <stmt> <stmts>
<stmt> → <assign> | <write> | <read> | <blocks>
<assign> → <var> := <expr> ;
<write> → Write (<varlist>);
<read> → Read (<varlist>);
<varlist> → <var> | <var> , <varlist>
<var> → <ident>
<ident> → <char> | <char> <ident>
<expr> → <expr> + <term> | <expr> **-** <term> | <term>
<term> → <term> * <factor> | <term> **/** <factor> | <factor>
<factor> → (<expr>) | <var> | <sinteger>
<datatypes> → <datatype> | <datatype> <datatypes>
<datatype> → <vartype> | <consttype>
<vartype> → VAR <varlist> : <varlast> ;
<varlast> → <varkind> |
ARRAY [<sinteger> .. <sinteger>] OF <varkind>;
<varkind> → INTEGER | REAL | BOOLEAN | CHAR
<consttype> → CONST <ident> = <value>;
<value> → '<string>' | <sinteger> | <real>
<string> → <char> | <char> <string>
<real> → <sinteger> . <integer>
<sinteger> → <sign> <integer>
<sign> → + | - | NULL
<integer> → <digit> | <digit> <integer>
<digit> → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<char> → A | B | C | … | Z | a | b | . . . | z

This grammar will allow the generation of programs like:

```
GOODPROG
PROG_START;
        DEFINE
                VAR A, B      :       INTEGER;
                VAR C         :       REAL;
                VAR F         :       ARRAY [-2..4] OF CHAR;
                CONST         CI      =       2.71828;
                CONST         CII     =       -23;
        END;
        BEGIN;
                B       :=      A;
                C       :=      CI;
                DEFINE        VAR     A : REAL;      END;
                BEGIN;
                        A       :=      B;
                END;
        END;
PROG_END;
```

II. **Your Program**

Assuming static scope rules; Develop an interpreter for the ***FirLan* ++** language. The interpreter should read a ***FirLan* ++** source program from a file and write its output to an output file. The interpreter should write into the output file the results of excuting a statement only if it is syntacticaly correct. Otherwise, in case of a syntax error, your program should write into the output file the erronous line along with an error message indicating the type of error.