The Tiny Language

Note: (Task (1) deliverable: you will deliver a document containing the RE rules of Tiny Language + DFA + Scanner)

A program in TINY consists of a set of functions (any number of functions and ends with a main function), each function is a sequence of statements including (declaration, assignment, write, read, if, repeat, function, comment, …) each statement consists of (number, string, identifier, expression, condition, …).

Language described as:

1) Number: any sequence of digits and maybe floats (e.g. 123 | 554 | 205 | 0.23 | …)
2) String: starts with double quotes followed by any combination of characters and digits then ends with double quotes (e.g. “Hello” | “2nd + 3rd” | …)
3) Reserved_Keywords: int | float | string | read | write | repeat | until | if | elseif | else | then | return | endl
4) Comment_Statement: starts with /* followed by any combination of characters and digits then ends with */ (e.g. /*this is a comment*/ | …)
5) Identifiers: starts with letter then any combination of letters and digits. (e.g. x | val | counter1 | str1 | s2 | …)
6) Function_Call: starts with Identifier then left bracket “(“ followed by zero or more Identifier separated by “,” and ends with right bracket “)”. (e.g. sum(a,b) | factorial(c) | rand() | …)
7) Term: maybe Number or Identifier or function call. (e.g. 441 | var1 | sum(a,b) | …)
8) Arithmatic_Operator: any arithmetic operation (+ | - | * | /)

9) Equation: starts with Term or left bracket “(“ followed by one or more Arithmatic_Operator and Term. with right bracket “)” for each left bracket (e.g. 3+5 | x +1 | (2+3)*10 | …)
10) Expression: may be a String, Term or Equation (e.g. “hi” | counter | 404 | 2+3 | …)
11) Assignment_Statement: starts with Identifier then assignment operator “:=” followed by Expression (e.g. x := 1 | y:= 2+3 | z := 2+3*2+(2-3)/1 | …)
12) Datatype: set of reserved keywords (int, float, string)
13) Declaration_Statement: starts with Datatype then one or more identifiers (assignment statement might exist) separated by coma and ends with semi-colon. (e.g. int x; | float x1,x2:=1,xy:=3; | …)
14) Write_Statement: starts with reserved keyword “write” followed by an Expression or endl and ends with semi-colon (e.g. write x; | write 5; | write 3+5; | write “Hello World”; | …)
15) Read_Statement: starts with reserved keyword “read” followed by an Identifier and ends with semi-colon (e.g. read x; | …)
16) Return_Statement: starts with reserved keyword “return” followed by Expression then ends with semi-colon (e.g. return a+b; | return 5; | return “Hi”; | …)
17) Condition_Operator: (less than “<” | greater than “>” | is equal “=” | not equal “<>”)
18) Condition: starts with Identifier then Condition_Operator then Term (e.g. z1 <> 10)
19) Boolean_Operator: AND operator “&&” and OR operator “||”
20) Condition_Statement: starts with Condition followed by zero or more Boolean_Operator and Condition (e.g. x < 5 && x > 1)

21) If_Statement: starts with reserved keyword “if” followed by Condition_Statement then reserved keyword “then” followed by set of Statements (i.e. any type of statement: write, read, assignment, declaration, …) then Else_If_Statment or Else_Statment or reserved keyword “end”
22) Else_If_Statement: same as if statement but starts with reserved keyword “elseif”
23) Else_Statement: starts with reserved keyword “else” followed by a set of Statements then ends with reserved keyword “end”
24) Repeat_Statement: starts with reserved keyword “repeat” followed by a set of Statements then reserved keyword “until” followed by Condition_Statement
25) FunctionName: same as Identifier
26) Parameter: starts with Datatype followed by Identifier (e.g. int x)
27) Function_Declaration: starts with Datatype followed by FunctionName followed by “(“ then zero or more Parameter separated by “,” then “)” (e.g. int sum(int a, int b) | …)
28) Function_Body: starts with curly bracket “{” then a set of Statements followed by Return_Statement and ends with “}”
29) Function_Statement: starts with Function_Declaration followed by Function_Body
30) Main_Function: starts with Datatype followed by reserved keyword “main” then “()” followed by Function_Body
31) Program: has zero or more Function_Statement followed by Main_Function

Code Sample

```
/*Sample program includes all 30 rules*/

int sum(int a, int b)
{

        return a + b;

}
int main()
{
        int val, counter;
        read val;
        counter:=0;
        repeat
                val := val - 1;
                write "Iteration number [";
                write counter;
                write "] the value of x = ";
                write val;
                write endl;
                counter := counter+1;
        until val = 1
        write endl;
        string s := "number of Iterations = ";
        write s;
        counter:=counter-1;
        write counter;
        /* complicated equation */
        float z1 := 3*2*(2+1)/2-5.3;
        z1 := z1 + sum(1,y);
        if  z1 > 5 || z1 < counter && z1 = 1 then
                write z1;
        elseif z1 < 5 then
                z1 := 5;
        else
        z1 := counter;
    end
    return 0;
}
```

Code Sample

```
/* Sample program in Tiny language – computes factorial*/

int main()
{

    int x;

    read x; /*input an integer*/

    if x > 0 then /*don't compute if x <= 0 */

        int fact := 1;

        repeat

            fact := fact * x;

            x := x – 1;

        until x = 0

        write fact; /*output factorial of x*/

    end

    return 0;

}
```