Faculty of Computer & Information Sciences
Ain Shams University
Subject: Compiler Theory
Year: 3rd level undergraduate
Academic year: 1st term 2021-2022

Milestone (2)-Task (1)-CFG Rules

Extra regular expressions:

Digit:=[0-9]

Letter:=[A-Za-z]

Statements:=
(Condition_Statement|Comment_Statement|Function_Call|Read_Statement| Return_Statement|

If_Statement|Else_Statement|Else_If_Statement|

Repeat_Statement| Write_Statement|

Assignment_Statement|Declaration_Statement)

Statements → Condition_Statement | Comment_Statement

| Function_Call | Read_Statement | If_Statement

| Else_Statement | Else_If_Statement

| Repeat_Statement | Write_Statement

| Assignment_Statement |Declaration_Statement

| Return_Statement

States → Statements States_repeatition

States_repeatition → Statements States_repeatition | ε

term := term MultOp Factor | Factor

term → Factor Ter

Ter → MultOp Factor Ter | ε

Factor → (Equation) | Term

AddOp := + | -

AddOp → + | -

MultOp := * | /

MultOp → * | /

****

1)Number:=^Digit+(\.Digit+)?

2)String:=^”(.*)”$

3)Reserved_Keywords:= (int | float | string | read | write | repeat | until | if | elseif | else | then | return | endl | end)

4)Comment_Statement:=^(\/ *) (.*)(*\/)$

5)Identifier:=Letter(Letter|Digit)*

6) Arithmetic_Operator:= (+ | - | * | /)

7)Boolean_Operator := (&& | \|\|)

8)Condition_Operator := (< | > | = | <>)

9)Function_Call:=Function_Name\((Expression(,Expression)*)?\)$

Function_Call → Function_Name (ArgList)

ArgList → Expression Arguments | ε

Arguments →, ArgList | ε

10)Term:=(Number|Identifier|Function_Call)

Term → Number|Identifier|Function_Call

Left Factoring: Term → Number|Identifier D

D-> ε | (ArgList)

11) *Equation := Equation AddOp term| term*

Equation → term Equ

Equ → AddOp term Equ | ε

12) Expression := (String|Term|Equation)

Expression → String|Term|Equation

13) Assignment_Statement := Identifier \:\= Expression

Assignment_Statement → Identifier := Expression

12) Datatype := (int|float|string)

Datatype → int|float|string

13) Declaration_Statement :=

^Datatype (identifier | Assignment_Statement)

(, identifier |, Assignment Statement)*;$

Declaration_Statement ⟶

Datatype (identifier | Assignment_Statement) Declare_A ;

Declare_A ⟶ ,(identifier | Assignment_Statement) Declare_A| ε

14)Write_Statement:=^write (Expression | endl) ;$

Write_Statement ⟶ write (Expression | endl);

15)Read_Statement:=^read (Identifier) ;$

Read_Statement ⟶ read Identifier ;

16)Return_Statement := ^return (Expression);$

Return_Statement ⟶ return Expression;

17)Condition := ^(Identifier) (Condition_Operator) (Term)

Condition ⟶ Identifier Condition_Operator Ter

18)Condition_Statement := ^(Condition)
[(Boolean_Operator)(Condition)]*

Condition_Statement ⟶ Condition Condition_State

Condition_State ⟶ Boolean_Operator Condition Condition_State | ε

19) FunctionName:= Identifier

FunctionName ⟶ Identifier

20)If_Statement := ^if (Condition_Statement) then

[Statements]+ (Else_If_Statement | Else_Statement| end)

If_Statement ⟶ if (Condition_Statement) Then States (Else_If_Statement | Else_Statement| end)

21)Else_If_Statement := ^elseif (Condition_Statement) then [Statements]+ (Else_If_Statement | Else_Statement | end)

Else_If_Statement ⟶ elseif (Condition_Statement) Then States (Else_If_Statement | Else_Statement| end)

22)Else_Statement := ^else [Statements]+ end$

Else_Statement ⟶ else States end

23)Repeat_Statements := ^repeat [Statements]+ until (Condition_Statement)

Repeat_Statements ⟶ repeat States until Condition_Statement

24) Parameter:= (Datatype)(Identifier)

Parameter ⟶ Datatype Identifier

25) Function_Declaration:= (Datatype)(Identifier)

\((Parameter(\,Parameter)*)? \)

Function_Declaration⟶ Datatype Identifier (Parameters_list)

Parameter_list ⟶ Parameter Parameter_repeatition | ε

Parameter_repeatition ->, Parameter Parameter_repeatition | ε

28) Function_Body:= \{ [Statements]+ (Return_Statement)\}

Function_Body -> { States Return_Statement }

29)Function_Statement:=(Function_Declaration) (Function_Body)

Function_Statement -> Function_Declaration Function_Body

30)Main_Function := (Datatype) main \(\) (Function_Body)

Main_Function -> Datatype main ( ) Function_Body

31)Program:= (Function_Statement)* (Main_Function)

Program -> User_Function Main_Function

User_Function -> Function_Statement User_Function | ε