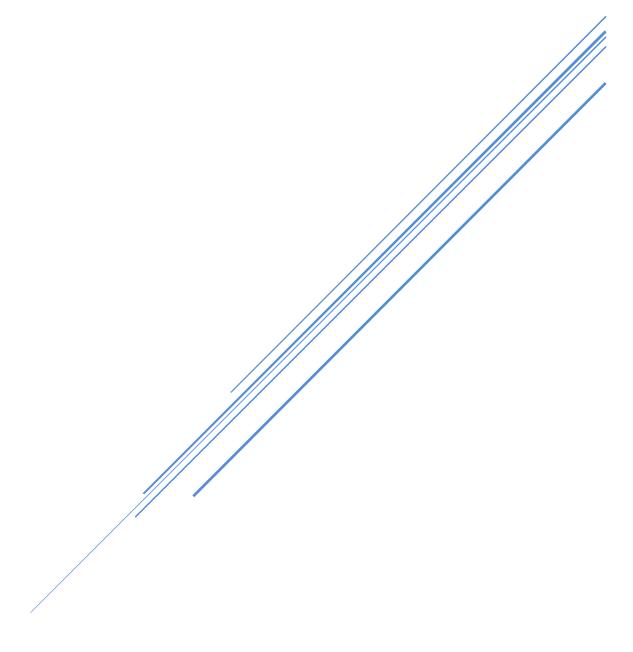
COMPILER THEORY

Tiny Compiler || MS2

Document containing Two Sections:

Section One : Terminals

Section Two: CFG



Terminals

Symbols and Operators

- { } (function body delimiters)
- () (for grouping and function parameters)
- ; (statement terminator)
- , (separator)
- := (assignment operator)
- +, (addition operators)
- *, / (multiplication operators)
- <, >, =, <> (relational operators)
- ss, || (Boolean operators)

Keywords

- int, float, string (datatypes)
- main (for main function)
- if, then, else, elseif, end (conditional statements)
- repeat, until (loop statements)
- read, write (I/O statements)
- return (function return)
- endl (end line)

Special Sequences

" (string delimiter)

Tokens

- identifier (user-defined names)
- number (numeric literals)
- letter (for identifiers)
- digit (for numbers)
- symbol (for comments)

CFG

Program Structure:

```
Program \rightarrow Functions Main_Function 
Functions \rightarrow Function_Statement Functions | \epsilon 
Main_Function \rightarrow int main () Function_Body
```

Function Declarations:

```
Function_Statement \rightarrow Function_Declaration Function_Body
Function_Declaration \rightarrow Datatype FunctionName (Parameters )
Parameters \rightarrow Parameter, Parameters | Parameter | \epsilon
Parameter \rightarrow Datatype identifier
FunctionName \rightarrow identifier
```

Function Body:

```
Function_Body → {Statements Return_Statement }
```

Statements:

```
Statements → Statement Statements | ε

Statement → Declaration_Statement

| Assignment_Statement

| Write_Statement

| Read_Statement

| If_Statement

| Repeat_Statement

| Function_Call;
```

```
Characters \rightarrow Character Characters | \epsilon Character \rightarrow letter | digit | symbol
```

Declaration Statements:

```
Declaration_Statement \rightarrow Datatype DeclarationList; DeclarationList \rightarrow DeclItem DeclarationList_Tail DeclarationList_Tail \rightarrow, DeclarationList | \epsilon DeclItem \rightarrow identifier DeclItem_Tail DeclItem_Tail \rightarrow := Expression | \epsilon Datatype \rightarrow int | float | string
```

Assignment Statements:

```
Assignment_Statement → identifier := Expression ;
```

Input/Output Statements:

```
Write_Statement → write Write_Tail
Write_Tail → Expression ; | endl ;
Read_Statement → read identifier ;
```

Conditional Statements:

```
If_Statement \rightarrow if Condition_Statement then Statements Else_Statement end Else_Statement \rightarrow elseif Condition_Statement then Statements Else_Statement | else Statements | \epsilon
```

Loops:

Repeat_Statement \rightarrow repeat Statements until Condition_Statement

Function Calls

```
Function_Call → identifier ( ArgumentList)
```

ArgumentList \rightarrow ExpressionList $\mid \epsilon \mid$

 ${\sf ExpressionList} \to {\sf ExpressionExpressionList_Tail}$

ExpressionList_Tail \rightarrow , ExpressionList | ϵ

Expressions:

```
Expression → String | Equation | Term
```

String \rightarrow " Characters "

Equation → Term Equation_Tail

Equation_Tail \rightarrow AddOp Equation | ϵ

 $\mathsf{Term} \to \mathsf{Factor}\,\mathsf{Term}_{-}\mathsf{Tail}$

Term_Tail \rightarrow MultOp Term | ϵ

Factor → number | identifier | Function_Call | (Equation)

Conditions:

 $Condition_Statement \rightarrow Condition\ Boolean Clause$

 $\label{eq:boolean_operator} \mbox{Boolean_Operator Condition_Statement} \ | \ \epsilon$ $\mbox{Condition} \rightarrow \mbox{Expression RelOp Expression}$

Operators:

AddOp
$$\rightarrow$$
 + | -

MultOp
$$\rightarrow * | /$$

$$\mathsf{RelOp} \mathbin{\rightarrow} \mathbin{<} |\mathbin{>}| = |\mathbin{<} \mathbin{>}$$

Boolean_Operator → && | ||

Return Statement:

Return_Statement \rightarrow return Expression;