Lexic.txt:

Alphabet:

a. Upper (A-Z) and lower case letters (a-z) of the English alphabet

b. Underline character '_';

c. Decimal digits (0-9);

a.Special symbols, representing:

- operators + - * /

- separators , . !

b.identifiers:

identifier ::= upper_case_letter | upper_case_letter{letter}{digit}

upper_case_letter ::= "A" | "B" | . ..| "Z"

letter ::= "A" | "B" | ... | "Z" | "a" | "b" | ... | "z"

digit ::= "0" | "1" |...| "9"

1.integer - rule:

noconst:="-"no|no

no:=digit{no}

2.character

character:='letter'|'digit'

3.string

constchar:="string"

string:=char{string}

char:=letter|digit

3.boolean

boolconst:="Truth" | "Lie"

Syntax.in:

program ::= "OnceUponATime" stmt "TheEnd"

stmt ::= declaration "." {stmt} | action "." {stmt}

declaration ::= IDENTIFIER "is" type

type1 ::= "Char" | "String" | "Integer" | "Boolean"

arraydecl ::= "Array" "of" type1

type ::= type1|arraydecl

action ::= simplstmt | structstmt

simplstmt ::= assignstmt | iostmt | footnotestmt

assignstmt ::= IDENTIFIER "equals" expression

expression ::= expression operator expression | expression

term ::= term "*" factor | factor

factor ::= "(" expression ")" | IDENTIFIER

operator ::= "+" | "-" | "/" | "*"

iostmt ::= "Hear" "(" IDENTIFIER ")" | "Speak" "(" {IDENTIFIER[,]} ")"

footnotestmt ::= "Footnote" string

structstmt ::= ifstmt | whilestmt

ifstmt ::= "If" condition "Procede" stmt ["Otherwise" stmt]

whilestmt ::= "Loop, until" condition "," stmt "!"

condition ::= expression [" is not "] RELATION expression

RELATION ::= "smaller" | "equal" | "greater"

Tokens.in:

Array, of, character, string, historical, Loop, If, Procede, Otherwise, int, Story, Hear, Speak, until, equal, equals, smaller, greater, is, not, OnceUponATime, TheEnd, footnote