CISC 603 Project Progress Report

Tianxiang Jin

tjin@my.harrisburgu.edu

Operator Grammar

- Expression – term ((+|-) term)*
- Term – element ((*|/) element)*
- Element – (+|-) element | integer | (expression)

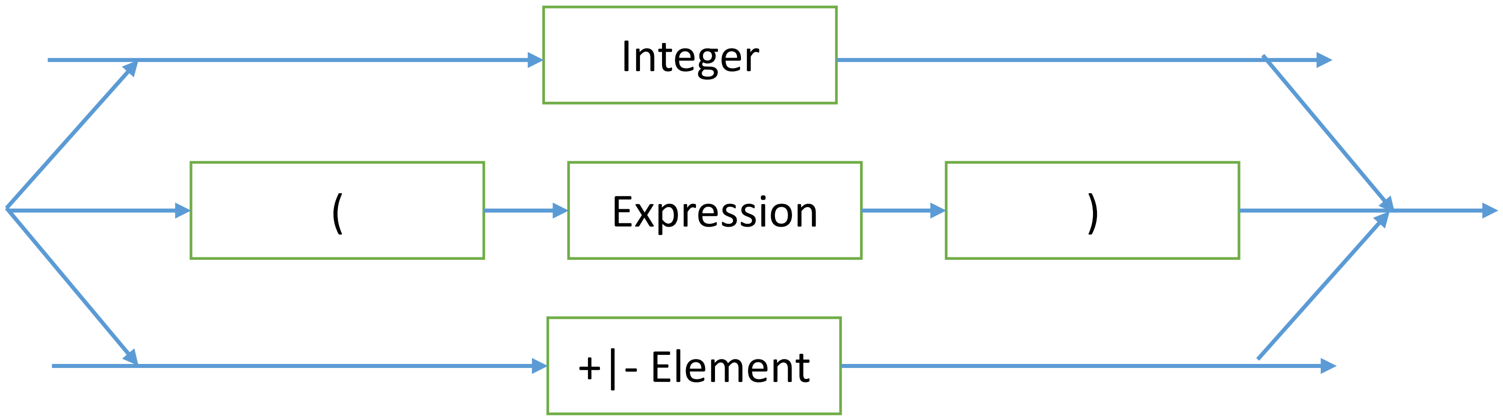
Expression

Term ((+|-) Term)*

Element + Element

Integer + Integer

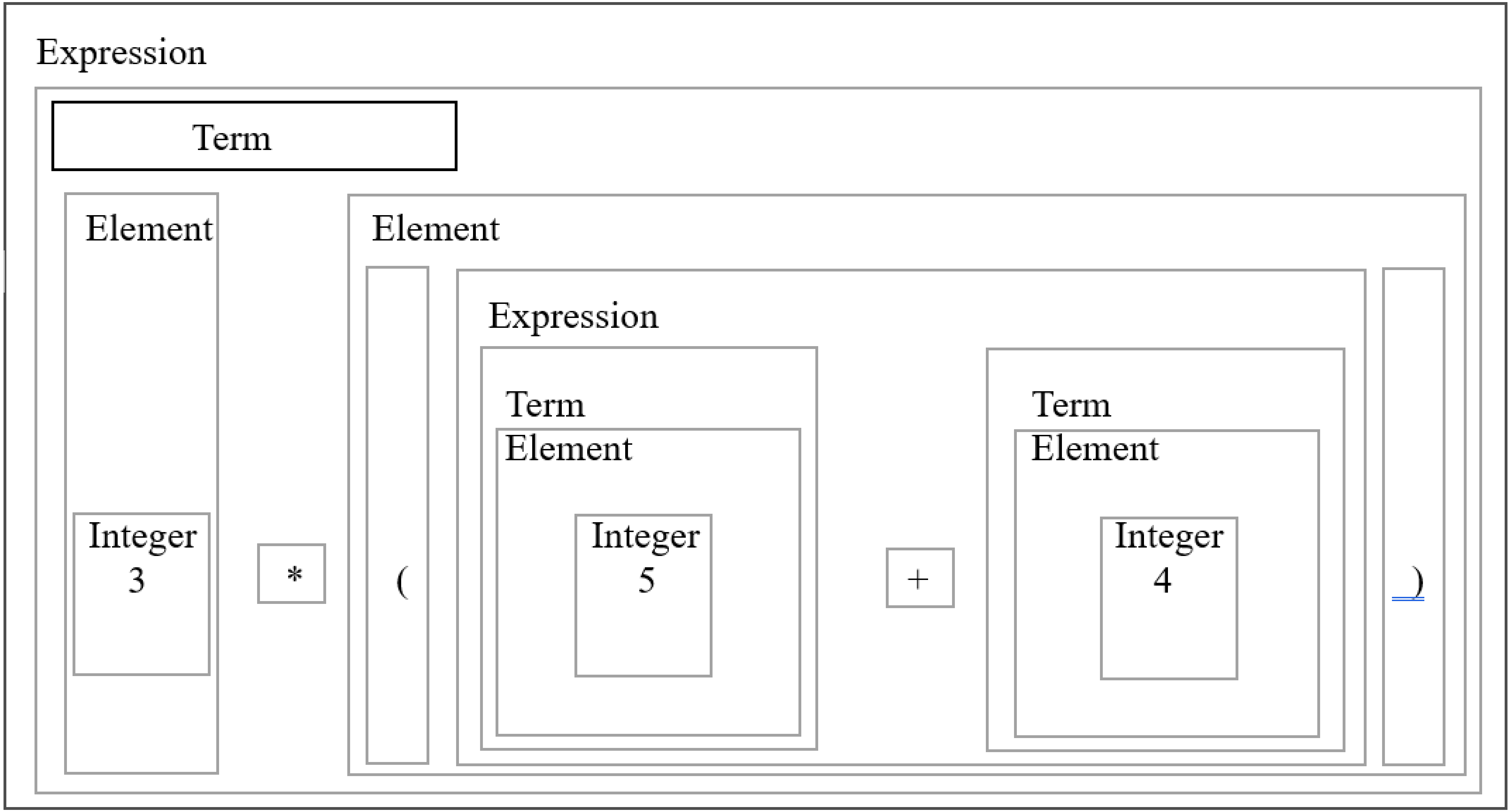
2 + 3



Integer

(Expression)

+|- Element


Expression
Term
Element
Integer
3
*
Element
(
Expression
Term
Element
Integer
5
+
Term
Element
Integer
4
)

keywords

Reserved Keywords including the following:

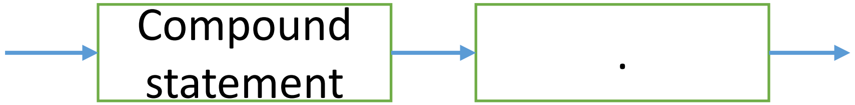
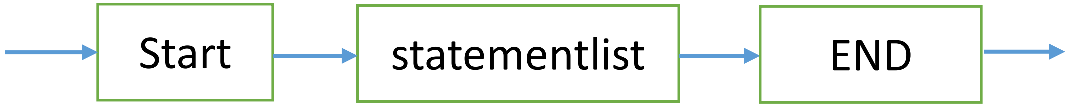
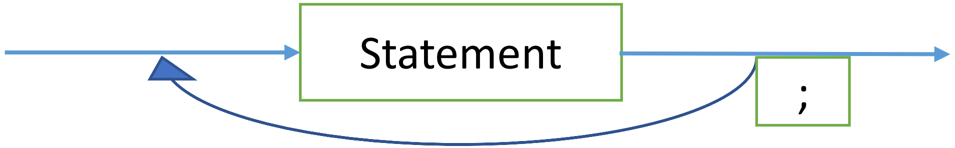
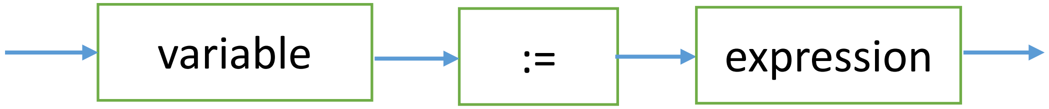
- START : Token(‘START’)
- END.: Token(‘END.’)
- VAR: Token(‘VAR’)
- INT: Token(‘INT’)
- FLOAT: Token(‘FLOAT’)
- CALCULATOR: Token(‘CALCULATOR’)
- “:=“, “;” ……

Symbol Table

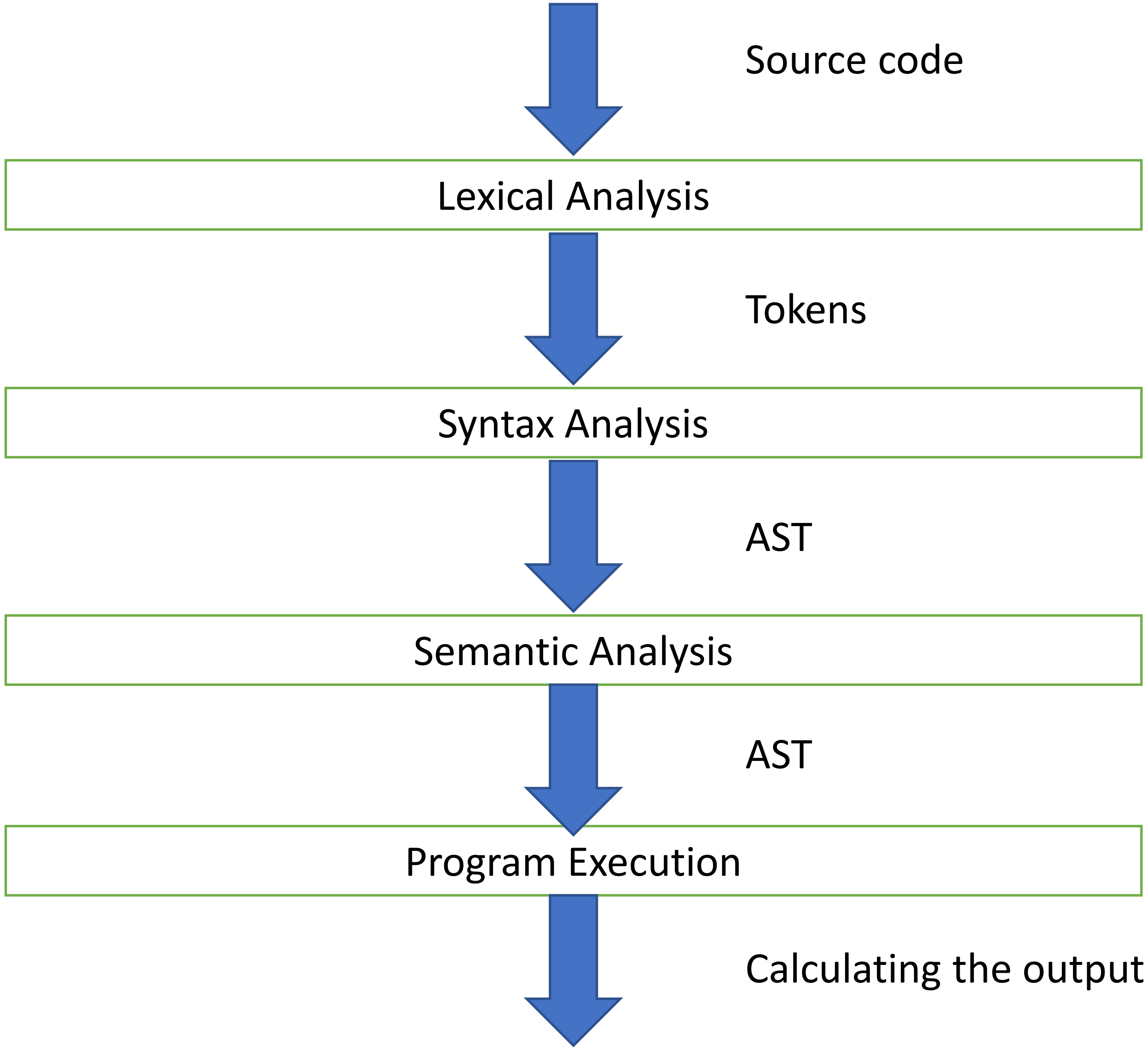
After a:= 3, b := 4

……	……
a	3
b	4

Program Grammar

Syntax Diagram	Grammar Rule
 Compound statement .	Calculator: compound statement .
 Start statementlist END	Compoundstatement: START statementlist END
 Statement ;	Statementlist: Statement | Statement ; Statement
	Statement: Compound statement | assignment statement | Empty
 variable := expression	Assignment statement: variable := expression
	Empty

Flow Chart



Source code

Lexical Analysis

Tokens

Syntax Analysis

AST

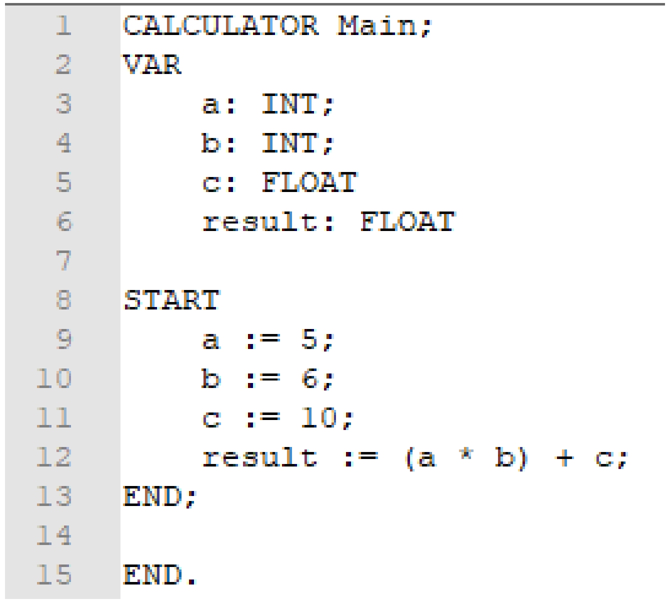
Semantic Analysis

AST

Program Execution

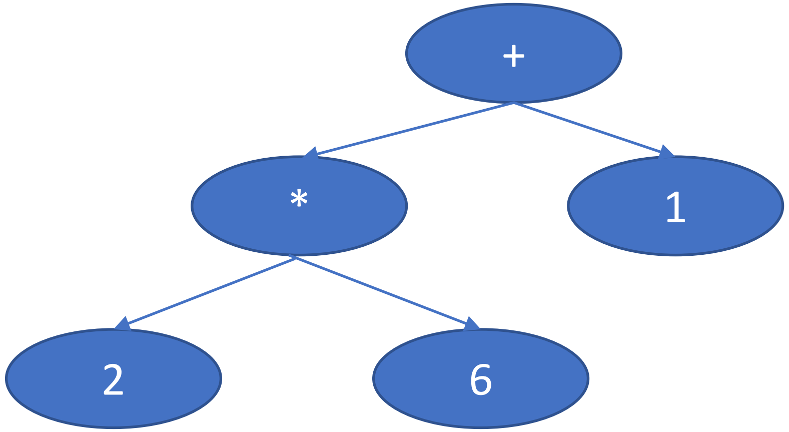
Calculating the output

Sample Code

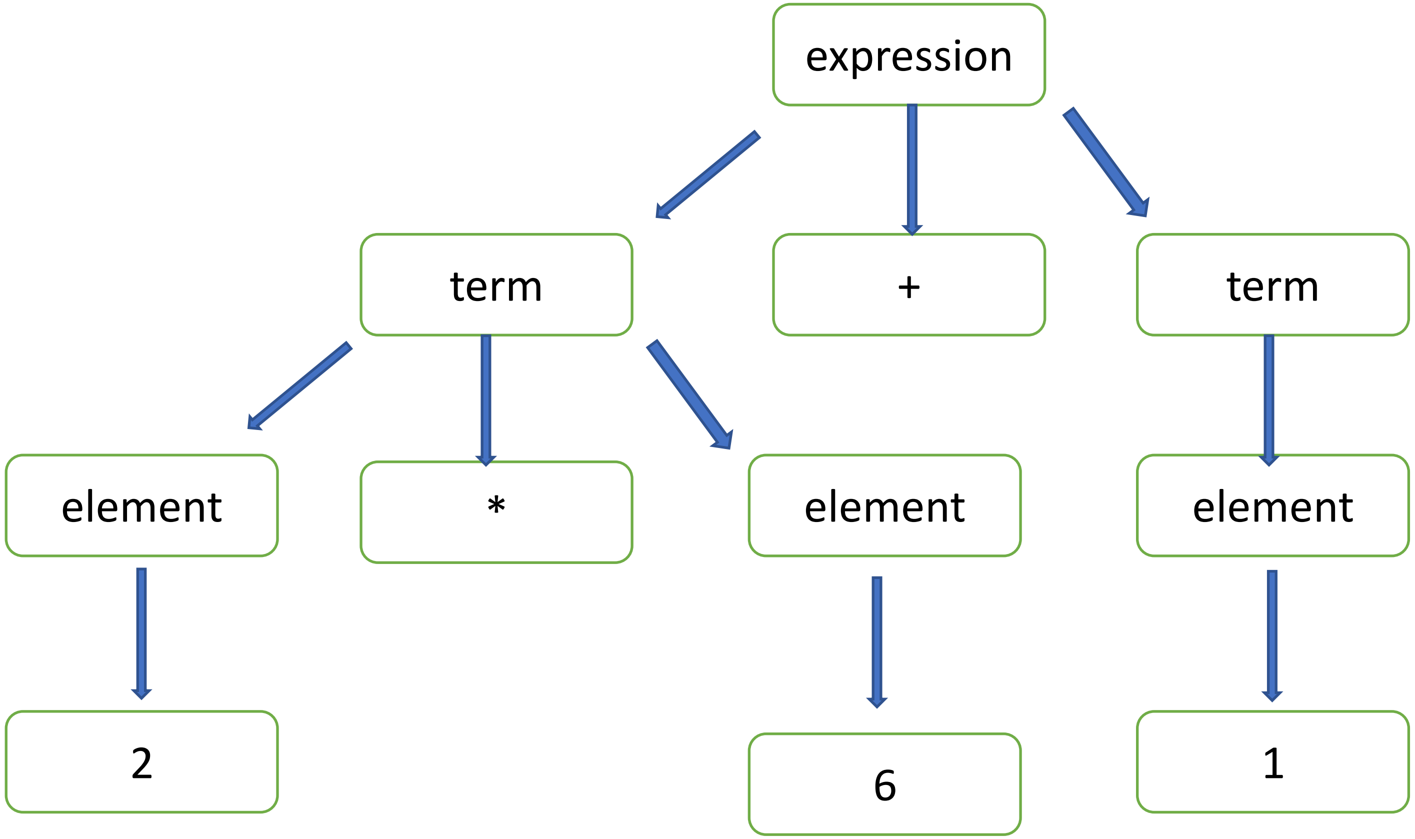


```
CALCULATOR Main;
VAR
    a: INT;
    b: INT;
    c: FLOAT
    result: FLOAT

START
    a := 5;
    b := 6;
    c := 10;
    result := (a * b) + c;
END;

END.
```

Sample AST



Parser Logic Tree



expression
term
+
term
element
*
element
element
2
6
1