Lab Three

Andrew Mathew

Andrew.Mathew2@Marist.edu

March 7, 2022

1 Crafting a Compiler

1.1 Exercise 4.7

(a) left-most derivation:
Start
E $
T plus E
F plus E $
num plus E $
num plus T plus E $
num plus T times F plus E $
num plus F times F plus E $
num plus num times F plus E $
num plus num times num plus E $
num plus num times num plus T $
num plus num times num plus F $
num plus num times num plus num $

(b) right-most derivation:
Start
E $
T plus E
T plus T $
T plus T times F $
T plus T times num $
T plus T times num $
T plus F times num $
T plus num times num $
T times F plus num times num $
T times num plus num times num $
F times num plus num times num $

num times num plus num times num $

(c) how the grammar structures expressions:
The ’times’ operator precedes the ’plus’ operator in this grammar.
The ’plus’ operator seems to have right associativity.

1.2 Exercise 5.2c

```
parseStart(){
  parseValue()
  match($)
}

parseValue(){   if Token is num
    match(num)
  else
    match(l_paren)
    parseExpr()
    match(r_paren)
  end if
}

parseExpr(){
  if Token is +
    match(+)
    parseValue()
    parseValue()
  else
    match(*)
    parseValues()
  end if
}
parseValues(){
  if Token in {num, (}
    parseValue()
    parseValues()
  else
    // do nothing when empty
  end if
}

match(expectedTokens){
  if currentToken in expectedTokens
    consume currentToken
    inc tokenPointer
  else
    error: expected expectedTokens but found currentToken
  end if }
```

2 Dragon

2.1 Exercise 4.2.1 a,b,c

(a) left-most

S
SS*
SS+S*
aS+S*
aa+S*
aa+a*

(b) right-most

S
SS*
Sa*
SS+a*
Sa+a*
aa+a*

(c)