Piradon (Tien) Liengtiraphan
Lab 2
Instructor: Dr. Alan G. Labouseur

CMPT 432L-111: Design of Compilers
Due: 2017-01-24

Lab 1

Exercise 4.7a. *A grammar for infix expressions follows (Fig 1):*

```
1 Start → E $
2 E     → T plus E
3       | T
4 T     → T times F
5       | F
6 F     → ( E )
7       | num
```

Show the leftmost derivation of the following string: ***num plus num times num plus num $***

Answer.

Leftmost Derivation
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 num plus E $
num plus num times num plus T $
num plus num times num plus F $
num plus num times num plus num $

Exercise 4.7b. *For Fig 1 show the rightmost derivation of the following string:* ***num plus num times num plus num $***

Answer.

Rightmost Derivation
E $
T plus E $
T plus T $
T plus T times F $
T plus T times num $
T plus F times num $
T plus num times num $
$ 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 $

Exercise 4.7c. *Describe how this grammar structures expression, in terms of precedence and left-or-right-associativity of operations.*

Answer.

In this particular grammar, expressions are defined to exit when followed to preceded by addition (plus) or multiplication (times) operations. Expressions are also composed of other expressions with the addition or multiplication operators, "num" being the objected acted upon.

Exercise 5.2c. *Construct a recursive-descent parser based on the grammar. (Fig 2)*

```
1 Start  → Value $
2 Value  → num
3        | lparen Expr rparen
4 Expr   → plus Value Value
5        | prod Values
6 Values → Value Values
7        | λ
```

Answer.

start: parseValue()

parseValue: match(num) or match(lparen)→parseExpr()→match(rparen)

parseExpr: match(plus)→parse(Value)→parse(Value) or match(prod)→parseValues()

parseValues: parse(Value)→parse(Value) or λ

match: Check if current token matches required argument, if yes, move to the next token.

Exercise 4.2.1a. *Consider the context-free grammar:*

$$S \to SS + |SS^*|a$$

and the string $aa + a^*$. *Give the leftmost derivation for the string.*

Answer.

$S = lm \Rightarrow SS + S^* \Rightarrow aS + S^* \Rightarrow aa + S^* \Rightarrow aa + a^*$

Exercise 4.2.1b. *Consider the context-free grammar:*

$$S \to SS + |SS^*|a$$

and the string $aa + a^*$. *Give the rightmost derivation for the string.*

Answer.

$S = rm \Rightarrow SS^* \Rightarrow Sa^* \Rightarrow SS + a^* \Rightarrow Sa + a^* \Rightarrow aa + a^*3$

Exercise 4.2.1c. *Consider the context-free grammar:*

$$S \to SS + |SS^*|a$$

and the string $aa + a^*$. *Give a parse tree for the string.*

Answer.