```
Author: Zachary Smith

Where [e] is equivalent to the empty set, or epsilon.

LL(1) Grammar =================================================================

1. <expr>   ::= <a> <t>
2. <t>      ::= <expr>
3.           | [e]
4. <a>      ::= <b> <u>
5. <u>      ::= <binop> <a>
6.           | [e]
7. <b>      ::= <incrop> <b>
8.           | <c>
9. <c>      ::= <d> <v>
10 <v>      ::= <incrop> <v>
11           | [e]
12 <d>      ::= $<b>
13           | <e>
14 <e>      ::= (<expr>)
15           | <num>
<incrop> ::= ++ | --
<binop>  ::= + | -
<num>    ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

NULLABLE ======================================================================

Nullable(expr)   = False
Nullable(t)      = True
Nullable(a)      = False
Nullable(u)      = True
Nullable(b)      = False
Nullable(c)      = False
Nullable(v)      = True
Nullable(d)      = False
Nullable(e)      = False
Nullable(incrop) = False
Nullable(binop)  = False
Nullable(num)    = False

FIRST =========================================================================

FIRST(expr)   = {incrop, F, (, num}
FIRST(t)      = {incrop, F, (, num, [e]}
FIRST(a)      = {incrop, F, (, num}
FIRST(u)      = {binop, [e]}
FIRST(b)      = {incrop, F, (, num}
FIRST(c)      = {F, (, num}
FIRST(v)      = {incrop}
FIRST(d)      = {F, (, num}
FIRST(e)      = {(, num}
FIRST(incrop) = {incrop}
FIRST(binop)  = {binop}
FIRST(num)    = {num}

FOLLOW ========================================================================

FOLLOW(expr)   = {), $}
FOLLOW(t)      = {), $}
FOLLOW(a)      = {num, ), $}
FOLLOW(u)      = {num, ), $}
FOLLOW(b)      = {incrop, binop, num, ), $}
```

FOLLOW(c)   = {incrop, binop, num,), $}
FOLLOW(v)   = {incrop, binop, num,), $}
FOLLOW(d)   = {incrop, binop, num,), $}
FOLLOW(e)   = {incrop, binop, num,), $}
FOLLOW(incrop) = {incrop, F, (, num,), $}
FOLLOW(binop) = {incrop, F, (, num}
FOLLOW(num)   = {incrop, binop, num,), $}

TABLE ==

	Input Symbols						
	(	)	num	incrop	binop	F	$
expr	r1		r1	r1		r1	
t	r2	r3	r2	r2		r2	r3
a	r4		r4	r4		r4	
u		r6	r6		r5		r6
b	r8		r8	r7		r8	
c	r9		r9			r9	
v		r11	r11	r10	r11		r11
d	r13		r13			r12	
e	r14		r14				

==

Is our grammar a LL(1) grammar?
- Yes. For each set of productions for each symbol, their FIRST() definitions are pairwise disjoint. Also, for each symbol, A, having a nullable RHS as a production, each other non-nullable FIRST(RHS) /\ FOLLOW(A) is disjoint. These fulfill the rules of LL(1) grammars. Furthermore, there is no left-recursion or ambiguity.