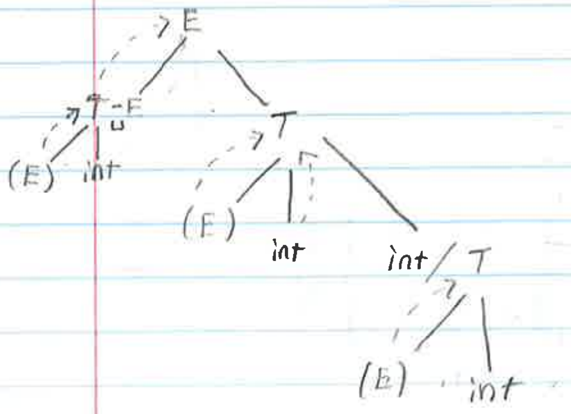
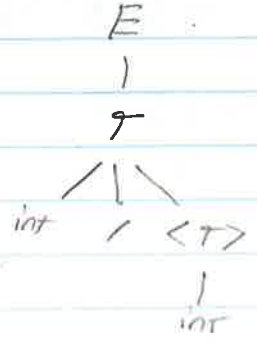
Homework #4

1. Do a Brute Force search to find a parse tree for the input int/int. Use top-down approach meaning you begin with the start rule (the first rule and find a left derivation. When choosing which rule to use in a derivation go in order of the rules. The int terminal is an integer literal

<E> ::= <T> - <E> | <T>
<T> ::= (<E>) | int | int / <T>

<E> → <T> - <E> → int - <E> X
→ <T> → (<E>) X
→ int X
→ int / <T> → int / (<E>) X
→ int / int ✓





② Compute the FIRST sets for the following. Compute follow sets as well for extra credit.

<A> ::= <A><A> '+' | <A><A> '*' | a

First (A) = {a}
First (+) = {+}
First (*) = {*}
First (a) = {a}

Follow (A) = {a, +, *, $}

(3) I

S → 0 S 1
S → 0 1

S → 0 y
y → S 1
y → 1

	0	1	$
S	0Y		
Y	S1	1	

II S → + S S | * S S | a

	+	*	a	$
S	+SS	-SS	a	

III

S → S(S) S | λ

S → (S)S
→ λ

	(	)	$
S	(S)S	λ	λ

(4) <X> ::= (<P>)
<P> ::= <Z><P> | <Z>
<Z> ::= 0 | 1

X → (P)
P → Z Y
Y → P
→ λ
Z → 0
→ 1

	0	1	(	)	$
X			(P)		
P	ZY	ZY			
Y	P	P		λ	
Z	0	1			

5. The Following is a grammar describes a language for regular expressions over symbols a and b; the language uses + -sign in place of a |-sign for union. |-sign is part of BNF Syntax, + part language

<rexpr> ::= <rexpr> '+' <rterm> | <r term>

<rterm> ::= <rterm> <r factor> | <r factor>

<rfactor> ::= <r factor> * | <r primary>

<r primary> ::= a | b

rexpr → rterm X

X → '+' rterm X

→ λ

r term → rfactor y

y → rfactor y

→ λ

rfactor → rprimary Z

Z → * Z

→ λ

rprimary → a

→ b

	+	'*'	a	b	$
<r expr>			rterm X	r term X	
X	+ rterm X				λ
<r term>			rfactor	rfactor	
Y	λ		rfactor y	rfactor y	λ
<r factor>			rprimary Z	rprimary Z	
Z		* Z			λ
<r primary>			a	b	

first (rexpr) = {a, b}

first (X) = {+} ∪ {$}

first (r term) = {a, b}

first (Y) = {a, b} ∪ {+, $}

first (rfactor) = {a, b}

first (Z) = {*} ∪ {$}

first (rprimary) = {a} ∪ {b}