ASSIGNMENT-3 COMPILERS

(Itish Agarwal, 18CS30021)

29 Sep' 2020

Q1.

(a) Expression : a + b*c + (d* e)

Postfix : abc*+de*+

(b) Infix expression : a-b-c

Postfix : ab-c-

(c) Infix expression : a*(b+c)/d

Postfix : abc+*d/

(d) Infix expression : (a+ (b*c))/(d-e)

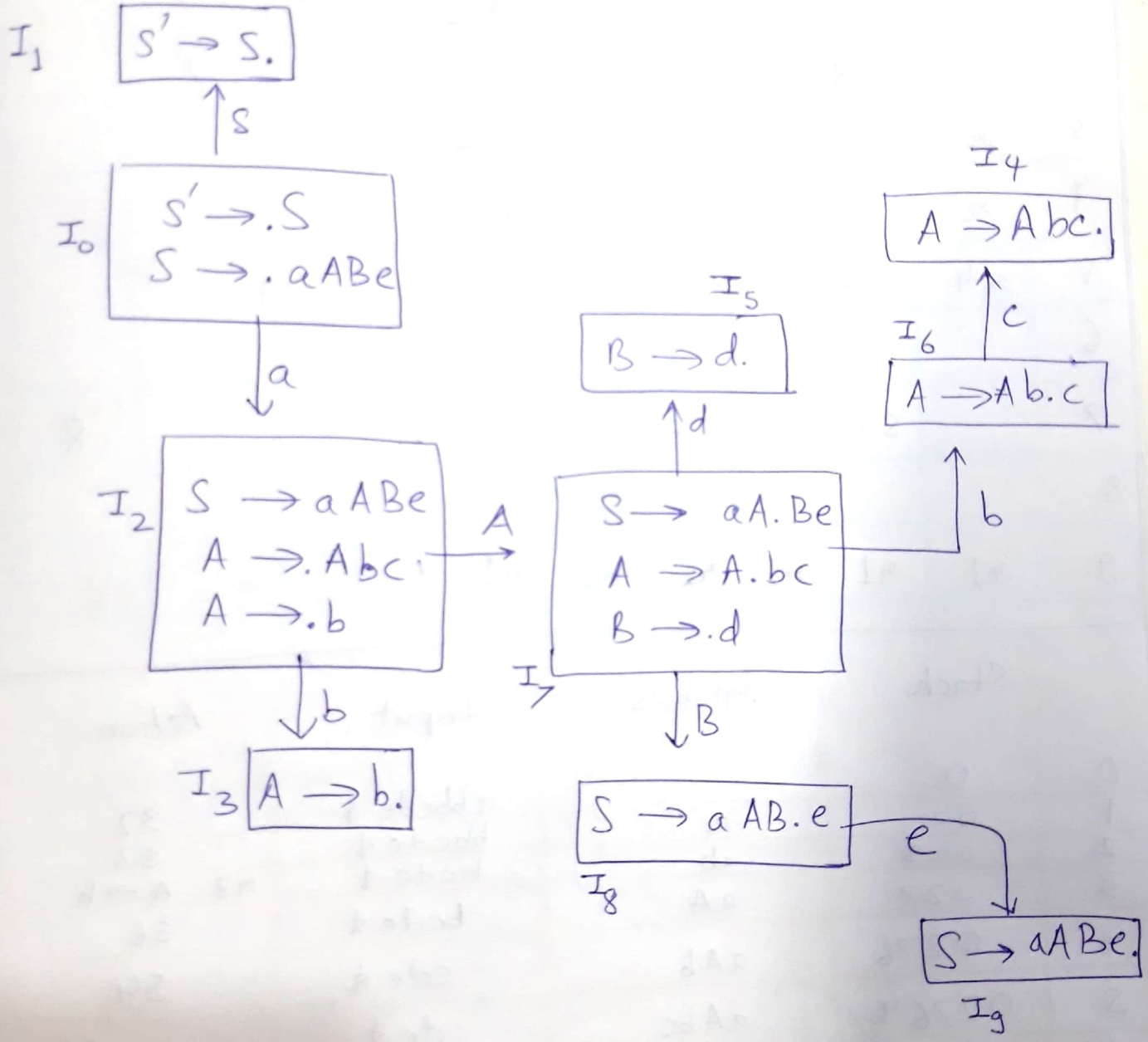
Postfix : abc*+de-/

Q2. Augmented Grammar :

S.NO	Production	S.No	Production
0 :	S' → S	3 :	A → b
1 :	S → a ABe	4 :	B → d
2 :	A → A bc		

Non-terminals : S, A, B

Terminals : a, b, c, d, e, $



PTO

	a	b	c	d	e	$	S	A	B
0	s2						1		
1						acc			
2		s3						7	
3	r3	r3	r3	r3	r3	r3			
4	r2	r2	r2	r2	r2	r2			
5	r4	r4	r4	r4	r4	r4			
6			s4						8
7		s6		s5					
8					s9				
9	r1	r1	r1	r1	r1	r1			

	Stack	Symbols	Input	Action
0	0		abbcde $	s2
1	02	a	bbcde $	s3
2	023	ab	bcde $	r3: A → b
3	027	aA	bcde $	s6
4	0276	aAb	cde $	s4
5	02764	aAbc	de $	r2: A → Abc
6	027	aA	de $	s5
7	0275	aAd	e $	r4: B → d
8	0278	aAB	e $	s9
9	02789	aABe	$	r1: S → aABe
10	01	S	$	accept

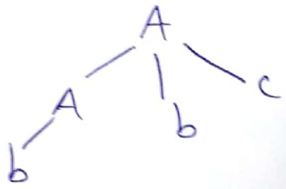
∴ The reduce actions that need to be applied to get a bottom-up parse tree:

(1) r3: A → b
(2) r2: A → Abc
(3) r4: B → d
(4) r1: S → aABe

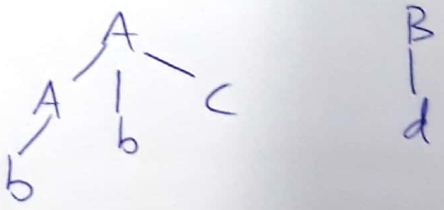
The parse-tree construction:

(1) A → b:

(2) A → Abc:



(3) B → d:



(4) S → aABe: