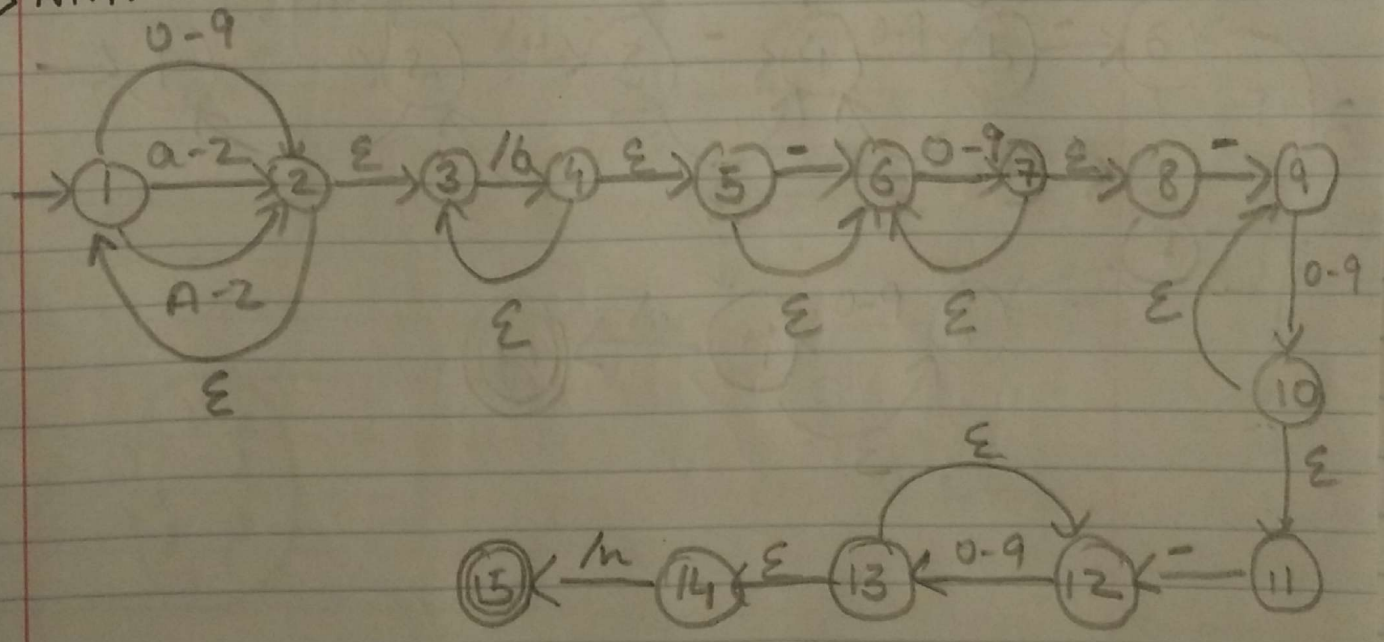
HW1

Name: KUMARI AASHA
USC-ID: 9488013423
565 - Compiler Design.

1)

a) $[a\text{-}z|A\text{-}z|0\text{-}9]+[/6]+[-]?[0\text{-}9]+[-][0\text{-}9][-][0\text{-}9]+[/n]$

b) NFA:

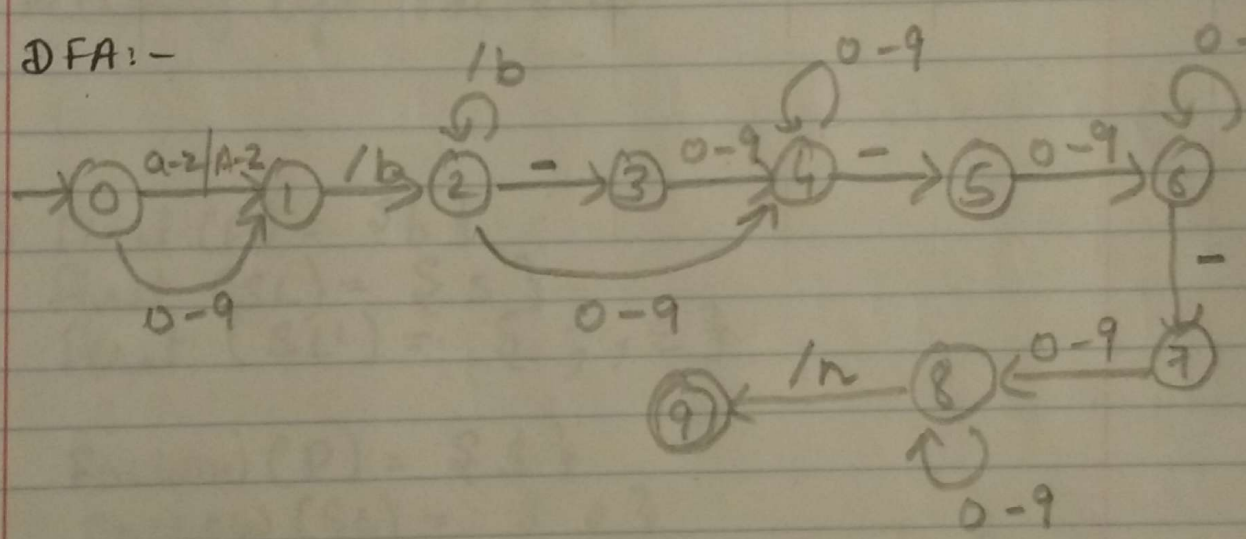


E → Error

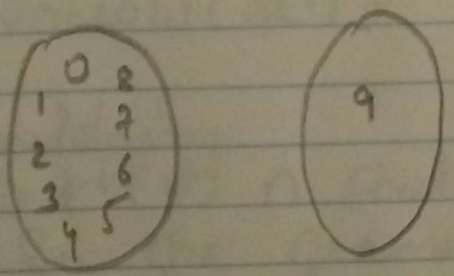
c)

DFA	NFA	a-z|A-Z	\b	-	0-9	\n
0	{1}	{1,2,3}	E	E	{1,2,3}	E
1	{1,2,3}	{1,2,3}	{3,4,5,6}	E	E	E
2	{3,4,5,6}	E	{3,4,5,6}	{6}	{6,7,8}	E
3	{6}	E	E	E	{6,7,8}	E
4	{6,7,8}	E	E	{9}	{6,7,8}	E
5	{9}	E	E	E	{9,10,11}	E
6	{9,10,11}	E	E	{12}	{9,10,11}	E
7	{12}	E	E	E	{12,13,14}	E
8	{12,13,14}	E	E	E	{12,13,14}	{15}
9	{15}	E	E	E	E	E

DFA :-

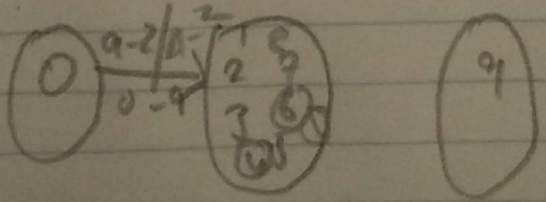


d)

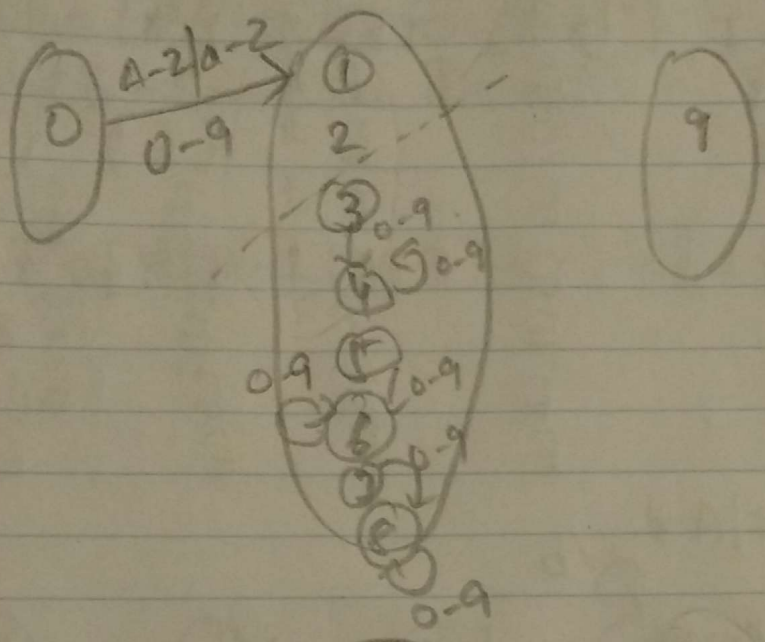


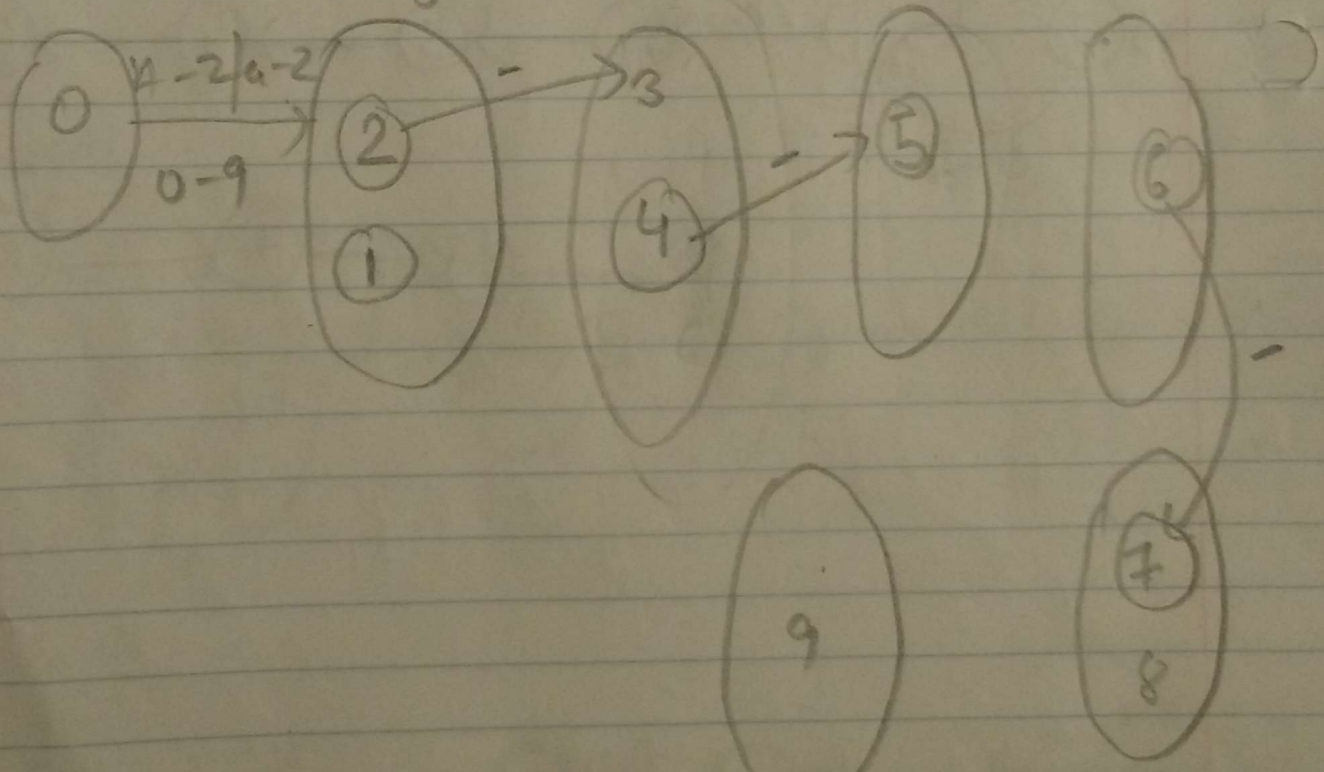
⇒ Initial state

Considering input a-z|A-Z|0-9 we get



Considering input -





Thereby each state will end up in a seperate partition.

e) #include <stdio.h>
#include <regex.h>

void regexp(char* string, char *pattern)
{
    regex_t regex;
    regmatch_t *match;
    char *pattern = "[a-z|A-Z|0-9]+ [/b]+[-]
                    [0-9]+[-][0-9]+[-][0-9]+
                    [\n]";
    if (regexec(&regex, str, match, 0))
    {
        printf("ERROR")
    }
    else
        matchnreplace(&string);
}

int main()
{ char str[];
    puts("Enter string:");
    gets(*str);
    regexp(&str);
}

void matchnreplace(char *string)
{
    i=find index of (' ')
    str1 = string.substr(0,i)
    printf('%s', str1)
// Similarly find index of last ' ' and replace by '), and do appropriate print }

2.)

a) $P \rightarrow b\ SL\ e$
$SL \rightarrow SL;s \mid s$

Removing left recursion

$P \rightarrow b\ SL\ e$
$SL \rightarrow s\ SL'$
$SL' \rightarrow ;\ s\ SL' \mid \varepsilon$

Already left factored

b) First (P) = {b}
First (SL) = {s}
First (SL') = {;, ε}

FOLLOW (P) = {$}
FOLLOW (SL) = {e}
FOLLOW (SL') = FOLLOW (SL) = {e}

Considering
First+(P) ∩ First+(SL) ∩ First+(SL') = ϕ
So the grammer is LL(1)

c)

	b	e	s	;	$
P	P→bSLe				
SL			SL→sSL'		
SL'		SL'→ε		SL'→;sSL'	

Stack	INPUT	OUTPUT
$P	bS;s;se$	
$eSLb	bS;s;se$	P→bSLe
$eSL	S;s;se$	SL→sSL'
$eSL's	S;s;se$	SL'→;sSL'
$eSL'	;s;se$	
$eSL's;	;s;se$	
$eSL's	s;se$	
$eSL'	;se$	SL'→;sSL'
$eSL's;	;se$	
$eSL's	se$	
$eSL'	e$	SL'→ε
$e	e$	
$	$	

"Matched"

Q3)

a) $E \to E+T$

$E \to T$

$T \to TF$

$T \to F$

$F \to F*$

$F \to a$

$F \to b$

First(E) = {a, b}
First(T) = {a, b}
First(F) = {a, b}

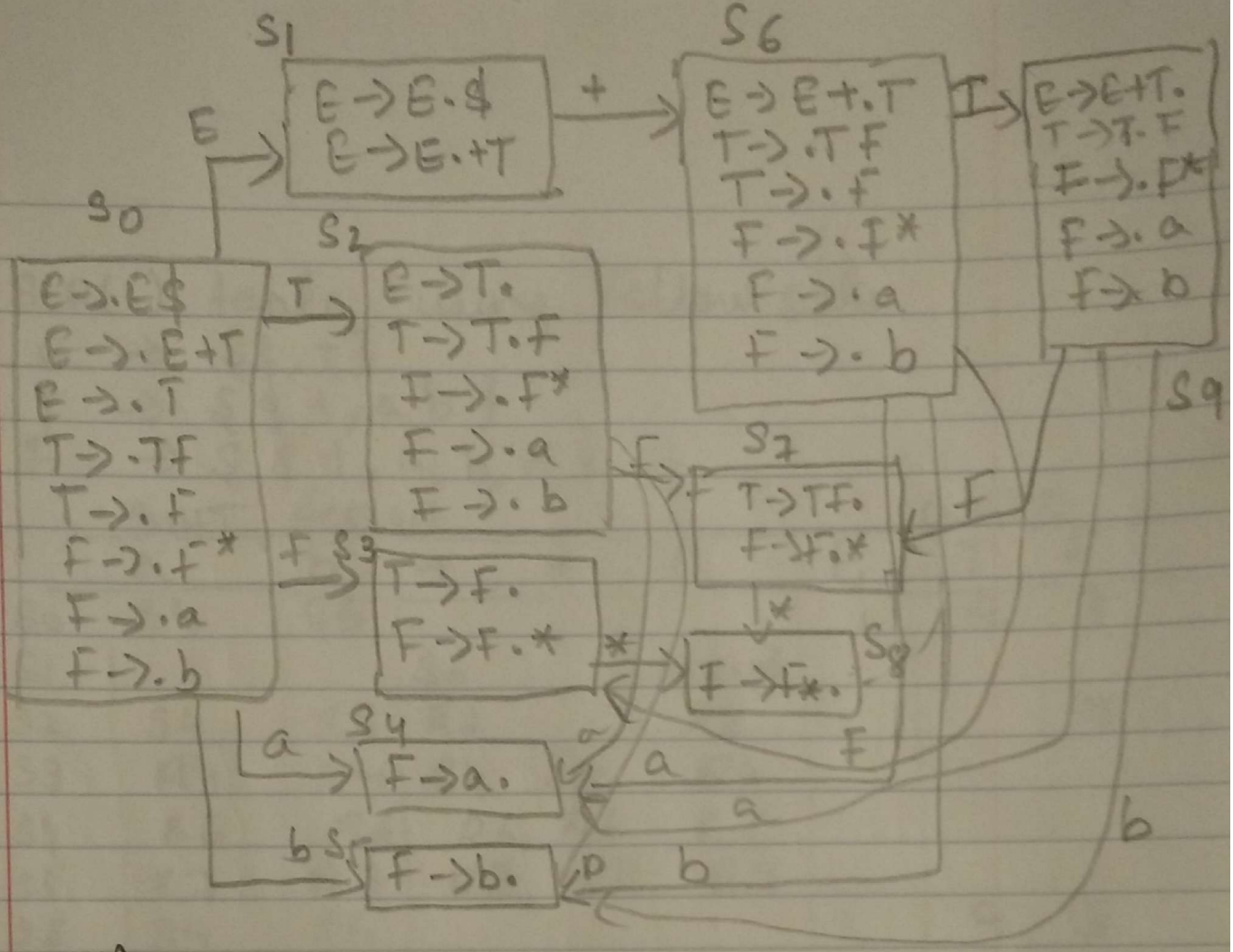
Follow(E) = {+, $}
Follow(T) = {a, b, +, $}
Follow(F) = {a, b, +, *, $}

b)

1. $E' \to E\$$
2. $E \to E+T$
3. $E \to T$
4. $T \to TF$
5. $F \to F*$
6. $F \to a$
7. $F \to b$

LR(0) items :- $E' \to .E\$$, $E' \to E.\$$

$E \to .E+T$
$E \to E.+T$
$E \to E+.T$
$E \to E+T.$
$E \to .T$
$E \to T.$
$T \to .TF$
$T \to T.F$
$T \to TF.$
$F \to .F*$
$F \to F.*$
$F \to F*.$
$F \to .a$
$F \to a.$
$F \to .b$
$F \to b.$



LR(0)

	SHIFT ACTION					GOTO		
	a	b	+	*	$	E	T	F
S0	S4	S5				1	2	3
S1			S6		Accept			
S2	S4/R2	S5/R2	R2	R2	R2			7
S3	R4	R4	R4	S8/R4	R4			
S4	R6	R6	R6	R6	R6			
S5	R7	R7	R7	R7	R7			
S6	S4	S5					9	3
S7	R3	R3	R3	S8/R3	R3			
S8	R5	R5	R5	R5	R5			
S9	S4/R1	S5/R1	R1	R1	R1			7

SLR(1) table using Follow sets

E → {$, +}
T → {$, +, a, b}
F → {$, +, a, b, *}

	SHIFT ACTION					GOTO		
	a	b	+	*	$	E	T	F
S0	S4	S5				1	2	3
S1			S6		Accept			
S2	S4	S5	R2		R2			7
S3	R4	R4	R4	S8	R4			
S4	R6	R6	R6	R6	R6			
S5	R7	R7	R7	R7	R7			
S6	S4	S5					9	3
S7	R3	R3	R3	S8	R3			
S8	R5	R5	R5	R5	R5			
S9	S4	S5	R1		R1			7

d) w = a+ab*$

Stack	Input	Action
0	a+ab*$	S4
0a4	+ab*$	R6 (F→a)
0F3	+ab*$	R4 (T→F)
0T2	+ab*$	S5
0T2b5	+ab*$	R7 (F→b)
0T2F7	+ab*$	R3 (T→TF)
0T2	+ab*$	R2 (E→T)
0E1	+ab*$	Accept