0.9
.9
9.

(1)

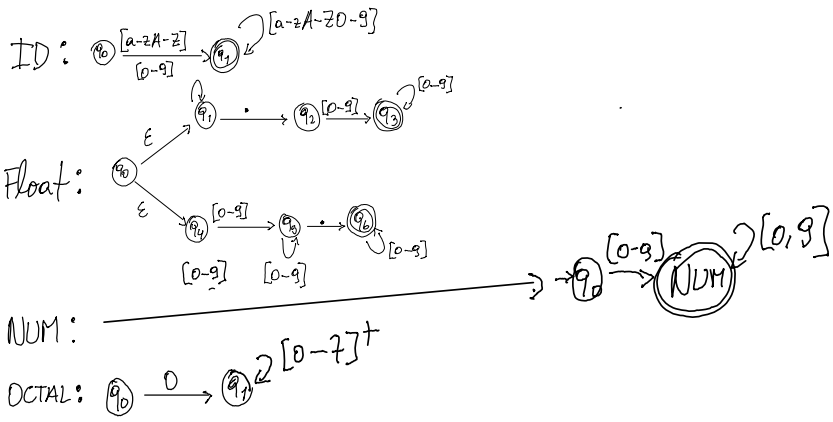
a) Regex ID: $[a-zA-Z]^+[a-zA-Z0-9]^*$

Regex Float: $[0-9]^*.[0-9]^+ \mid [0-9]^+.[0-9]^*$

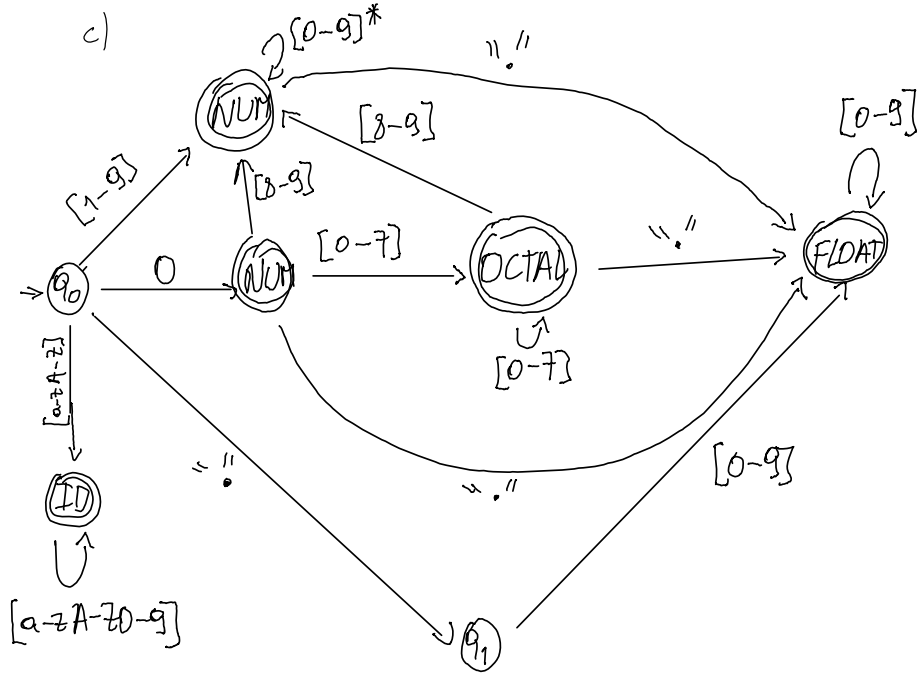
Regex Num: $[0-9]^+ \setminus \{OCTAL\}$

Regex Octal: $0[0-7]^+$

b)

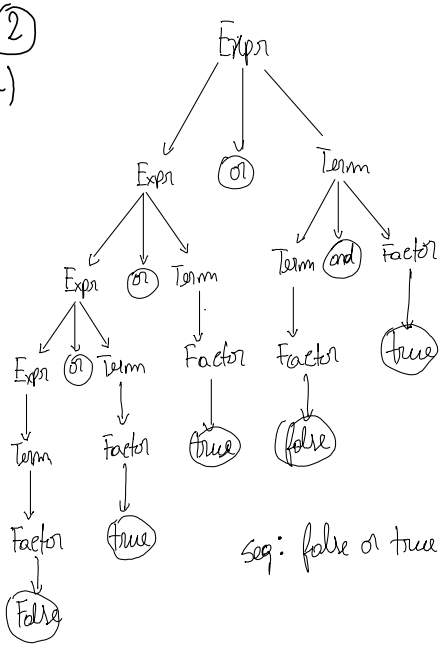


c)



(2)

a)



Seq: false or true or true or false and true

α sequência de símbolos (termos ou π)

First(α) conjunto de terminais que são os primeiros símbolos em uma ou mais sequências geradas por α
$\varepsilon \in$ First(α) se α for anulável

FOLLOW(A) conjunto de terminais que se podem seguir a A numa derivação sendo A não terminal.

(2)

b)

Cálculo dos conjuntos First:

First(Expr) = First(Term)

First(Term) = First(Factor)

First(Factor) = {true, false, (}

Cálculo dos conjuntos Follow

Follow(Expr) = {$} ∪ First()) \ ε ∪ First(or) \ ε = {$,), or}

Follow(Term) = First(and) \ ε ∪ Follow(Expr) =
= {and, or,), $}

Follow(Factor) = Follow(Term)

	or	and	ε	true	false	)	$
Expr				1, 2	1, 2		
Term				3, 4	3, 4		
Factor							

Para cada produção $A \to \alpha$

1. Para cada $t \in$ First(α) colocar $A \to \alpha$ em $T[A, t]$
2. Se $\varepsilon \in$ First(α), α é anulável, ⇒ para cada terminal $t \in$ FOLLOW(A), incluir $A \to \alpha$ em $T[A, t]$

Conflitos ⇒ Não é LL

c)

Gramática Inicial:

Expr → Expr or Term | Term β_1

Term → Term and Factor α_2 | Factor β_2

Factor → True | False | (Expr)

Nota (Eliminar LR):

$N \to N\alpha_1 \mid \dots \mid N\alpha_m \mid \beta_1 \mid \dots \mid \beta_n \iff N \to \beta_1 N' \mid \dots \mid \beta_n N'$, $N' \to \alpha_1 N' \mid \dots \mid \alpha_m N' \mid \varepsilon$

Remover Left Recursion:

(1) Expr → Term Expr'
(2) Expr' → or Term Expr'
(3) | ε
(4) Term → Factor Term'
(5) Term' → and Factor Term'
(6) | ε
(7) Factor → True
(8) | False
(9) | (Expr)

First(Expr) = First(Term) = First(Factor) = {true, false, (}

First(Factor) = {true, false, (}

First(Expr') = {or, ε}

First(Term) = First(Factor) = {true, false, (}

First(Term') = {and, ε}

Follow(Expr) = First())\ε ∪ {$} = {), $}

Follow(Expr') = Follow(Expr) = {), $}

Follow(Term) = First(Expr')\ε ∪ Follow(Expr) = {or,), $}

Follow(Term') = Follow(Term) = {or,), $}

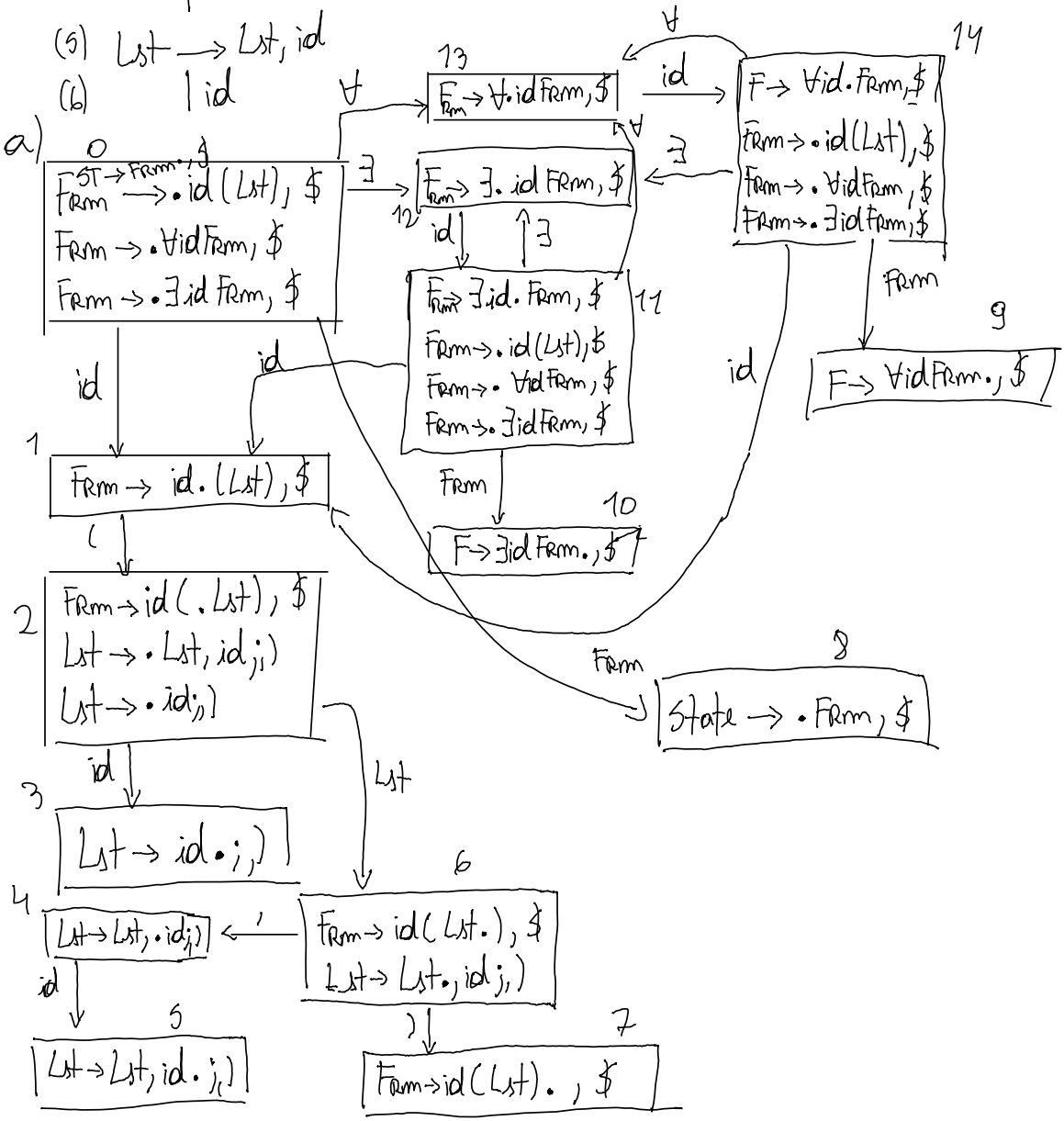
Follow(Factor) = First(Term')\ε ∪ Follow(Term) = {and, or,), $}

	and	or	true	false	(	)	$
Expr			1	1	1		
Expr'		2				2	3
Term			4	4	4		
Term'	5	6				5	6
Factor			7	8	9		

③ (1) ST → Frm $
(2) Frm → id (Lst)
(3) | ∀ id Frm
(4) | ∃ id Frm
(5) Lst → Lst , id
(6) | id

First(Frm) = {id} ∪ {∀} ∪ {∃} =
= {id, ∀, ∃}

First(Lst) = {id}



b)

	id	(	)	∀	∃	,	$	L	F
0	s1			s13	s12				G8
1		s2							
2	s3							g6	
3			R6			R6			
4	s5								
5			R5			R5			
6			s7			s4			
7							R2		
8							ACC		
9							R3		
10							R4		
11	s1								G10
12	s11								
13	s14								
14	s1			s13	s12				G9

e) ∀id ∃id id (id)

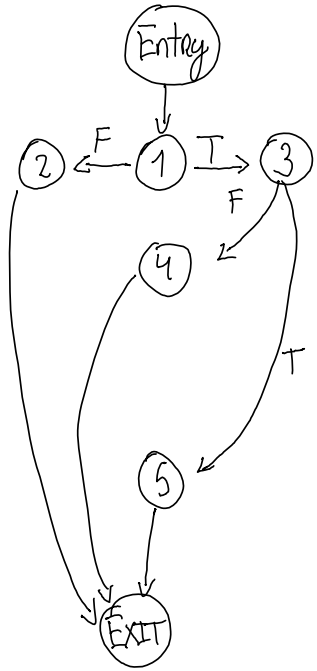
Stack	Current Head	Action
{}	∀id ∃id id (id) $	s13
{∀13}	id ∃id id (id) $	s14
{∀13, id14}	∃id id (id) $	s12
{∀13, id14, ∃12}	id id (id) $	s11
{∀13, id14, ∃12, id11}	id (id) $	s1
{∀13, id14, ∃12, id11, id1}	(id) $	s2
{∀13, id14, ∃12, id11, id1, (2}	id) $	13
{∀3, id14, ∃12, id11, id1, (2, id3}	) $	R6
{∀3, id14, ∃12, id11, id1, (2, 6	) $	s7
{∀3 id14 ∃12 id11, id1, (2, 6)4}	$	R2
∀3 id14 ∃12 id11, From10	$	R4
∀3 id14 From9	$	R3
F8	$	ACC

4a)

```
fib:                                   1
1- n := param[0]
2- if False n==1 goto L1
3- t1 := 1                             2
4- return t1
L1:                                    3
5- if False n==2 goto L2
6- t2 := 1                             4
7- return t2
L2:                                    5
8- t3 := n-2
9- t4 := n-1
10- param t3
11- t5 := call fib, 1
12- param t4
13- t6 := call fib, 1
14- t7 := t5+t6
15- return t7
main:
16- t8 := 4
17- param t8
18- t9 := call fib, 1
19- return t9
```

b)

Leaders = {1, 2, 5, 6, 8}



c)