

PL1

a) Parse tree for  $((a, a), a, (a))$ .

```

graph TD
    EXP1[EXP] --> L1["("]
    EXP1 --> LIST1[LIST]
    EXP1 --> R1[")"]
    LIST1 --> EST[EST]
    LIST1 --> COMMA[,]
    LIST1 --> EXP2[EXP]
    EXP2 --> L2["("]
    EXP2 --> LIST2[LIST]
    EXP2 --> R2[")"]
    LIST2 --> EXP3[EXP]
    EXP3 --> A[a]
  
```

$$\langle \text{EXP} \rangle ::= (\text{LIST}) \mid a$$

$$\langle \text{LIST} \rangle ::= \text{EXP} \{, \text{EXP} \}$$

c)

The diagram shows a linked list with two nodes. The first node contains the value 'r' and points to the second node. The second node contains the value '1' and points back to the first node, forming a cycle. A label 'EXP' is at the start of the first node's arrow, and a label 'LIST' is at the start of the second node's arrow.

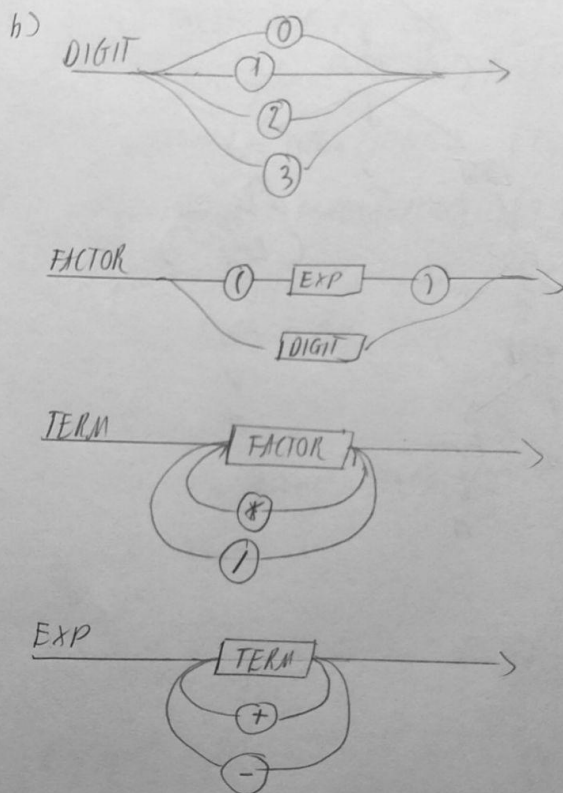
$$\text{First}(\text{LIST}) = \text{First}(\text{EXP}) = \{ (, q \}$$
$$\text{Follow}(\text{EXP}) = \{ , \} \cup \text{Follow}(\text{LIST}) = \{ , \} \cup \{ ) \} = \{ , , ) \}$$
$$\text{TERM} ::= \text{TERM} * \text{FACTOR} \mid \text{TERM} / \text{FACTOR} \mid \text{FACTOR}$$
$$\text{FACTOR} ::= (\text{EXP}) \mid \text{DIGIT}$$

DIGIT<sub>2</sub> = 0 / 1 / 2 / 3.

What Doan

2) a) to EBNF

$\langle \text{EXP} \rangle ::= \text{TERM} \{ (+|-) \text{TERM} \}$   
 $\langle \text{TERM} \rangle ::= \text{FACTOR} \{ (*|/) \text{FACTOR} \}$   
 $\langle \text{FACTOR} \rangle ::= (\text{EXP}) \mid \text{DIGIT}$   
 $\langle \text{DIGIT} \rangle ::= 0 \mid 1 \mid 2 \mid 3$



d)  $\text{First}(\text{DIGIT}) = \{0, 1, 2, 3\}$

$\text{First}(\text{FACTOR}) = \{ ( \cup \text{First}(\text{DIGIT}) \}$   
 $= \{ (, 0, 1, 2, 3 \}$

$\text{First}(\text{TERM}) = \text{First}(\text{FACTOR})$   
 $= \{ (, 0, 1, 2, 3 \}$

$\text{First}(\text{EXP}) = \text{First}(\text{TERM})$   
 $= \{ (, 0, 1, 2, 3 \}$

$\text{Follow}(\text{EXP}) = \{ ) \}$

$\text{Follow}(\text{TERM}) = \{ +, - \} \cup \text{Follow}(\text{EXP})$   
 $= \{ +, -, ) \}$

$\text{Follow}(\text{FACTOR}) = \{ *, / \} \cup \text{Follow}(\text{TERM})$   
 $= \{ *, /, +, -, ) \}$

$\text{Follow}(\text{DIGIT}) = \text{Follow}(\text{FACTOR})$   
 $= \{ *, /, +, -, ) \}$

c) Two requirements on a grammar for a predictive parser to be able to make right choice:

- Firstsets of any two choices must not have any token in common.
- If it is optional, then  $\text{First}(t) \cap \text{Follow}(t)$  should be empty.

What Do we

c) Prove that grammar satisfy the two requirements defined in (C).

Our First sets of any two choices must not have any token in common.

DIGIT is a trivial case.

$$\text{Factor} : \{ \{ \} \cap \text{First}(\text{DIGIT}) = \{ \{ \} \cap \{0, 1, 2, 3\} = \emptyset$$

TERM and EXP only have 1 first choice.

(2) If  $\epsilon$  is optional, then  $\text{First}(A) \cap \text{Follow}(A)$  should be empty.

$$\text{First}(\text{DIGIT}) \cap \text{Follow}(\text{DIGIT}) = \{0, 1, 2, 3\} \cap \{*, /, +, -, \epsilon\} = \emptyset$$

$$\text{First}(\text{FACTOR}) \cap \text{Follow}(\text{FACTOR}) = \{0, 1, 2, 3\} \cap \{*, /, +, -, \epsilon\} = \emptyset$$

$$\text{First}(\text{TERM}) \cap \text{Follow}(\text{TERM}) = \{+, -, \epsilon\} \cap \{*, /, +, -, \epsilon\} = \emptyset$$

$$\text{First}(\text{EXP}) \cap \text{Follow}(\text{EXP}) = \{0, 1, 2, 3\} \cap \{\epsilon\} = \emptyset$$

(3) Recursive Descent recognizer Pseudocode.

Procedure Start:

EXP

if token  $\Rightarrow$  error

Procedure EXP:

TERM

while (token is + or -)

{ match token

TERM }

Procedure TERM:

FACTOR

while (token is \* or /)

FACTOR }

Procedure FACTOR:

DIGIT

if (token is "(")

{ match token

DIGIT

match ")" }

else DIGIT

Procedure DIGIT

if (token is 1 or 2 or 3 or 4)

match token.

else \$error.

Procedure match(input)

if token match input

advance next token.

else \$error



Scanned with CamScanner