5

NFA/DFA AND LEX

1112914　宋冠穎
1112919　丁柏凱
1112929　詹佳瑜
1112946　王嘉汶
1112957　顏琦恩

OUTLINE

- 題目介紹 & 流程圖
- Regular Expression -> NFA
- NFA -> DFA
- DFA + DFA -> NFA -> DFA
- 程式
- Demo
- 結論
- 分工

題目

1 利用 Dotted Item Method

把以下的 Regular Expressions 轉 NFA/DFA

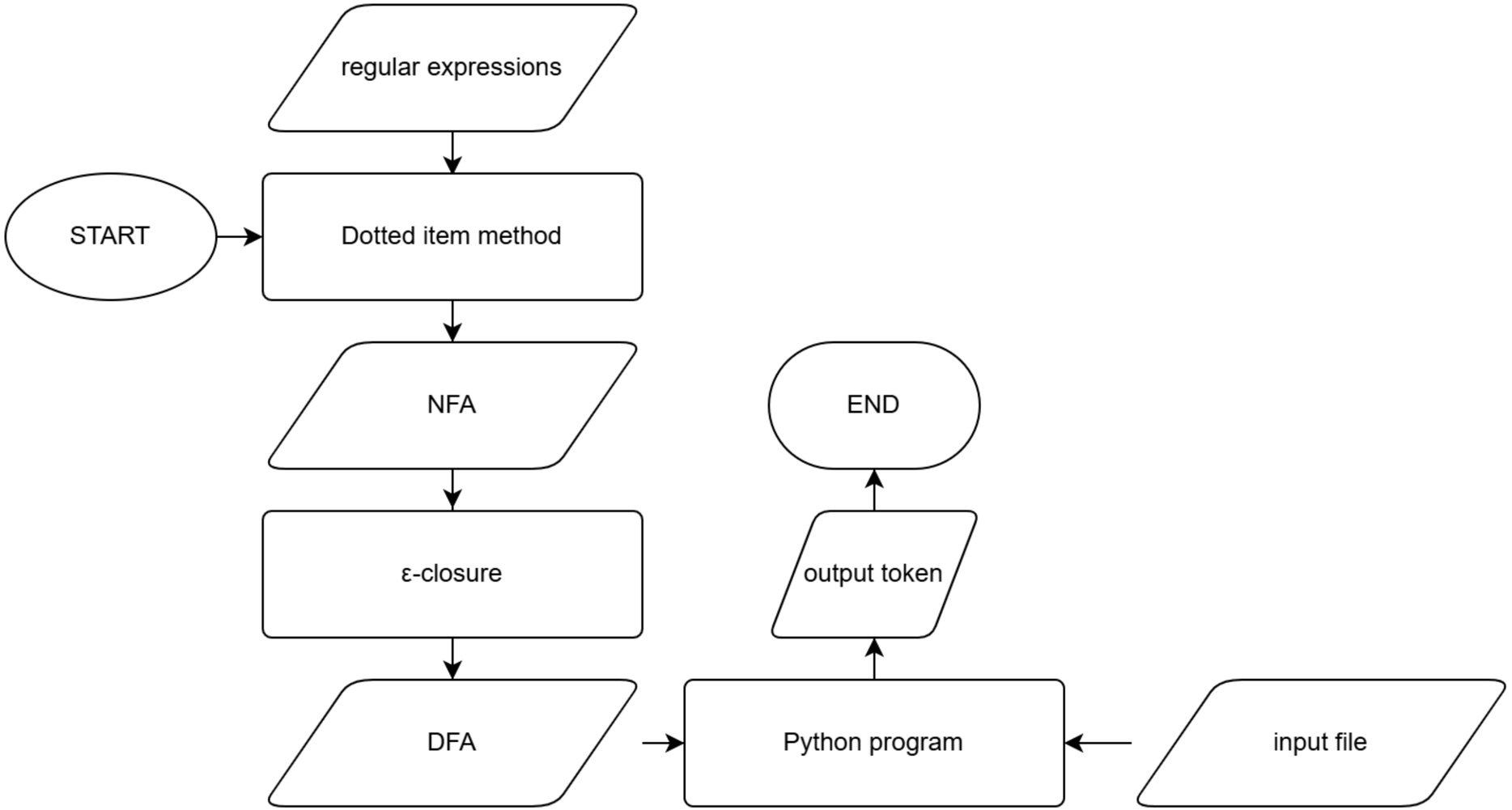
$$
\begin{aligned}
&\textbf{underscore} \rightarrow \_ \\
&\textbf{letter} \rightarrow \mathrm{A} \mid \mathrm{B} \ldots \mid \mathrm{Z} \mid \mathrm{a} \mid \mathrm{b} \ldots \mid \mathrm{z} \\
&\textbf{digit} \rightarrow 0 \mid 1 \mid 2 \ldots \mid 9 \\
&\textbf{underscore\_tail} \rightarrow \textbf{underscore}\ (\textbf{letter} \mid \textbf{digit})^{+} \\
&\textbf{id} \rightarrow \textbf{letter}\ (\textbf{letter} \mid \textbf{digit})^{*}\ \textbf{underscore\_tail}^{*} \\
&\textbf{fraction} \rightarrow \cdot\ \textbf{digit}^{+} \\
&\textbf{optional\_exponent} \rightarrow (\mathrm{E}\ (+ \mid - \mid \varepsilon)\ \textbf{digit}^{+}) \mid \varepsilon \\
&\textbf{integer} \rightarrow \textbf{digit}^{+} \\
&\textbf{float} \rightarrow \textbf{digit}^{*}\ \textbf{fraction}\ \textbf{optional\_exponent}
\end{aligned}
$$

2 寫一個 Lexical Analyzer 來分析讀到的是哪個 Token

Token 類型包含以下及id、integer、float

Separators and Brackets: the language uses ';', '(', ')', '[', ']', '{', '}' as the separators or brackets.
Binary Operators: the language uses the following operators +, -, *, /, <=, >=, <>, <, >, =, ==.
Keywords: the language uses the following keywords int, float, bool, void, while, if, else, for, return.
Comments: the language supports C style comments. A comment starts with /* and end with */ with anything in between except a comment ending sequence (*/).
Layout: the language uses the following layout symbols: any number of blank, tabulation, carriage return and line feed.

流程圖


regular expressions
START
Dotted item method
NFA
ε-closure
DFA
Python program
input file
output token
END

REGULAR EXPRESSION -> NFA

NFA介紹

- 可以對同一個輸入字元有多個轉移方向
- 可以存在 ε-transition（空轉移，不需輸入符號即可移動）

Dotted Item Method 介紹

- 帶有「dot ・」的產生式
- 標記目前已經分析到哪個位置
- dot 左側符號: 已經「看到」
- dot 右側符號: 將要分析

Dotted Item Method 介紹

$$
\begin{aligned}
A \rightarrow XYZ \quad &\Rightarrow A \rightarrow \cdot XYZ \\
&\Rightarrow A \rightarrow X\cdot YZ \\
&\Rightarrow A \rightarrow XY\cdot Z \\
&\Rightarrow A \rightarrow XYZ\cdot
\end{aligned}
$$

Dotted Item Method 介紹

- Character moves
  - 基本項目（Basic item）
    - 可以直接移動
    - ・A => A・
  - 非基本項目（Non-basic item）
    - 不可直接移動
    - (letter | digit)*

Dotted Item Method 介紹

- Non-basic item ε-move

$$T \to \alpha \bullet (R)^* \beta \Rightarrow T \to \alpha (R)^* \bullet \beta,\ T \to \alpha (\bullet R)^* \beta$$

$$T \to \alpha (R \bullet)^* \beta \Rightarrow T \to \alpha (R)^* \bullet \beta,\ T \to \alpha (\bullet R)^* \beta$$

$$T \to \alpha \bullet (R)^+ \beta \Rightarrow T \to \alpha (\bullet R)^+ \beta$$

$$T \to \alpha (R \bullet)^+ \beta \Rightarrow T \to \alpha (R)^+ \bullet \beta,\ T \to \alpha (\bullet R)^+ \beta$$

- dotted item 過程為簡化狀態機有進行部分簡化

⓪ ⇒① $underscore\_tail \rightarrow underscore \bullet (letter \mid digit)^{+}$

① ⇒② $underscore\_tail \rightarrow underscore(\bullet letter \mid digit)^{+}$

① ⇒③ $underscore\_tail \rightarrow underscore(letter \mid \bullet digit)^{+}$

② ⇒④ $underscore\_tail \rightarrow underscore(letter \mid digit)^{+} \bullet <<<<< recognized$

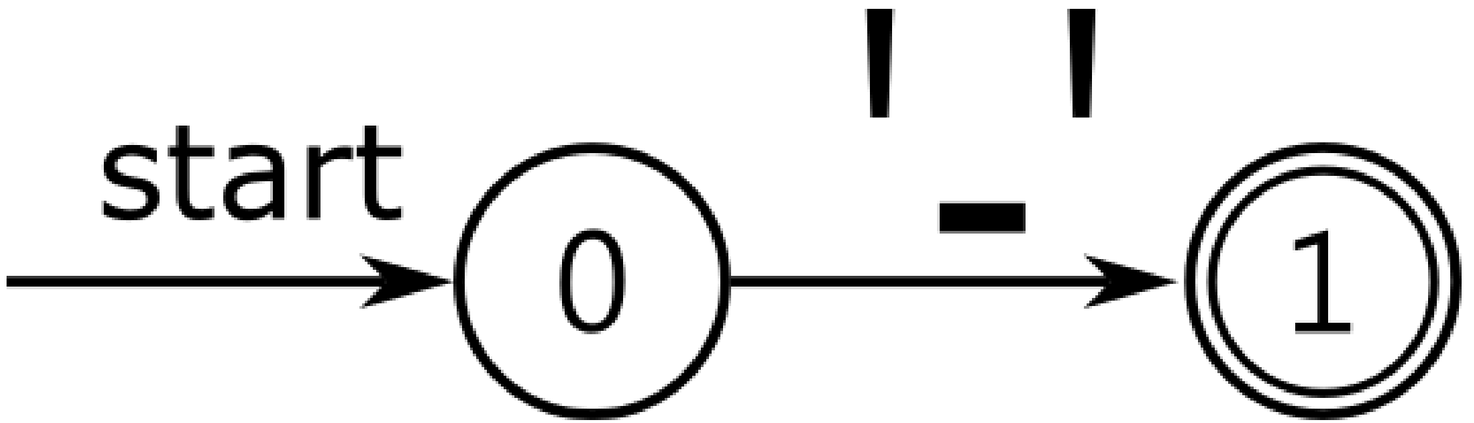
② ⇒① $underscore\_tail \rightarrow underscore \bullet (letter \mid digit)^{+}$

underscore

$$underscore \rightarrow \_$$

⓪ $underscore \rightarrow \bullet\_$

① $underscore \rightarrow \_\bullet$ $<<<<$ *recognized*

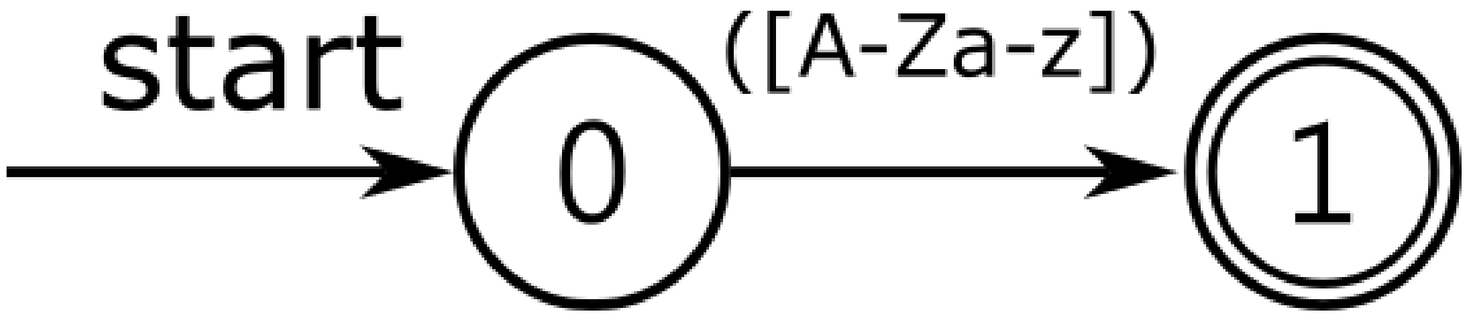


letter

$$letter \rightarrow [A - Za - z]$$

⓪ $letter \rightarrow \bullet[A - Za - z]$

⓪ ⇒ ① $letter \rightarrow [A - Za - z]\bullet$

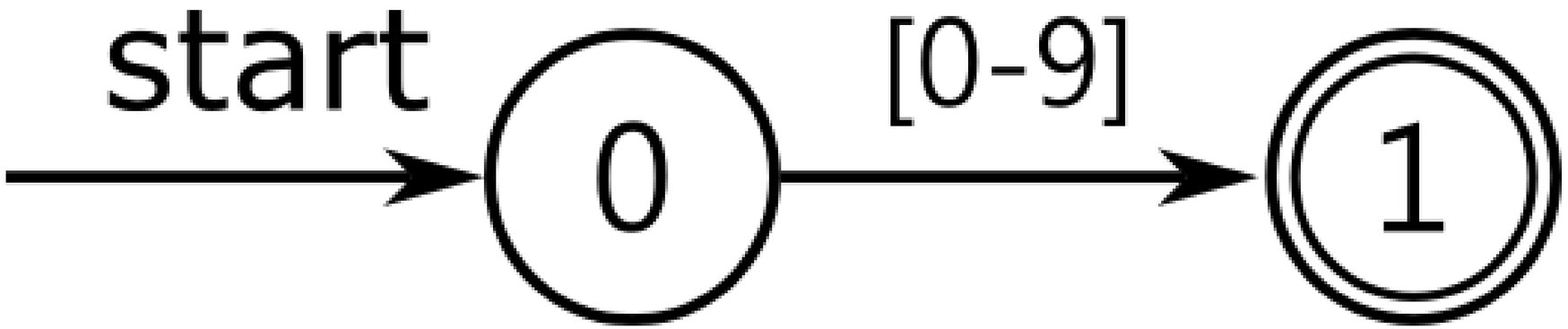


digit

$$digit \rightarrow [0-9]$$

⓪ $digit \rightarrow \bullet[0-9]$

⓪ ⇒① $digit \rightarrow [0-9]\bullet$ <<<<< *recognized*



underscore_tail

$$underscore\_tail \rightarrow underscore(letter \mid digit)^{+}$$

⓪ $underscore\_tail \rightarrow \bullet underscore(letter \mid digit)^{+}$

⓪ ⇒① $underscore\_tail \rightarrow underscore \bullet (letter \mid digit)^{+}$

① ⇒② $underscore\_tail \rightarrow underscore(\bullet letter \mid digit)^{+}$

① ⇒③ $underscore\_tail \rightarrow underscore(letter \mid \bullet digit)^{+}$

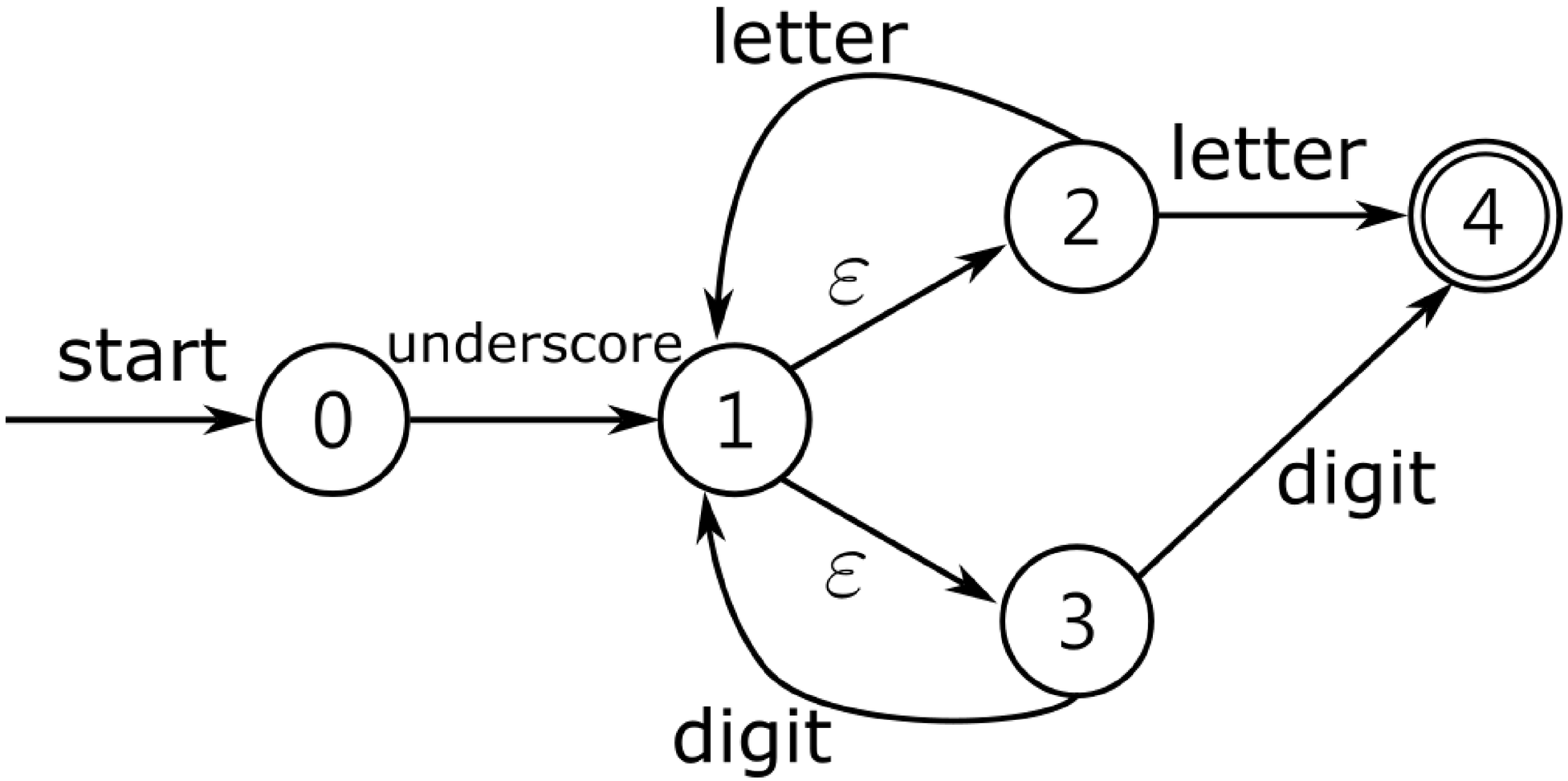
② ⇒④ $underscore\_tail \rightarrow underscore(letter \mid digit)^{+} \bullet <<<< recognized$

② ⇒① $underscore\_tail \rightarrow underscore \bullet (letter \mid digit)^{+}$

③ ⇒④ $underscore\_tail \rightarrow underscore(letter \mid digit)^{+} \bullet <<<< recognized$

③ ⇒① $underscore\_tail \rightarrow underscore \bullet (letter \mid digit)^{+}$

underscore_tail



id

$$id \rightarrow letter(letter \mid digit)^*(underscore\_tail)$$

⓪ $id \rightarrow \bullet letter(letter \mid digit)^*(underscore\_tail)$

⓪ ⇒① $id \rightarrow letter \bullet (letter \mid digit)^*(underscore\_tail)$

① ⇒② $id \rightarrow letter(\bullet letter \mid digit)^*(underscore\_tail)$
① ⇒③ $id \rightarrow letter(letter \mid \bullet digit)^*(underscore\_tail)$
① ⇒④ $id \rightarrow letter(letter \mid digit)^* \bullet (underscore\_tail)$

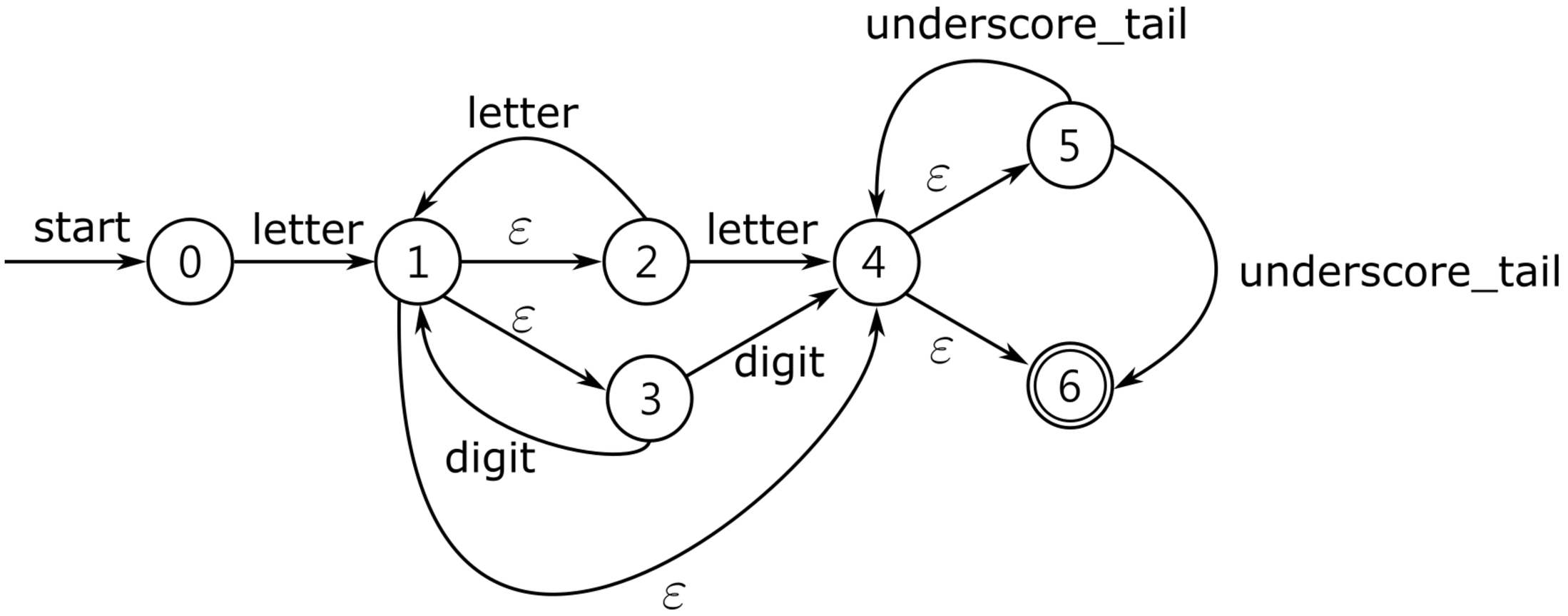
② ⇒④ $id \rightarrow letter(letter \mid digit)^* \bullet (underscore\_tail)$
② ⇒① $id \rightarrow letter \bullet (letter \mid digit)^*(underscore\_tail)$

③ ⇒④ $id \rightarrow letter(letter \mid digit)^* \bullet (underscore\_tail)$
③ ⇒① $id \rightarrow letter \bullet (letter \mid digit)^*(underscore\_tail)$

④ ⇒⑤ $id \rightarrow letter(letter \mid digit)^*(\bullet underscore\_tail)$
④ ⇒⑥ $id \rightarrow letter(letter \mid digit)^*(underscore\_tail)\bullet$ <<<< recognized

⑤ ⇒⑥ $id \rightarrow letter(letter \mid digit)^*(underscore\_tail)\bullet$ <<<< recognized
⑤ ⇒④ $id \rightarrow letter(letter \mid digit)^* \bullet (underscore\_tail)$

id


underscore_tail
letter
start
letter
ε
letter
ε
underscore_tail
digit
ε
digit
ε
0
1
2
3
4
5
6

fraction

$$fraction \rightarrow .(digit)^{+}$$

⓪ $fraction \rightarrow \bullet.(digit)^{+}$

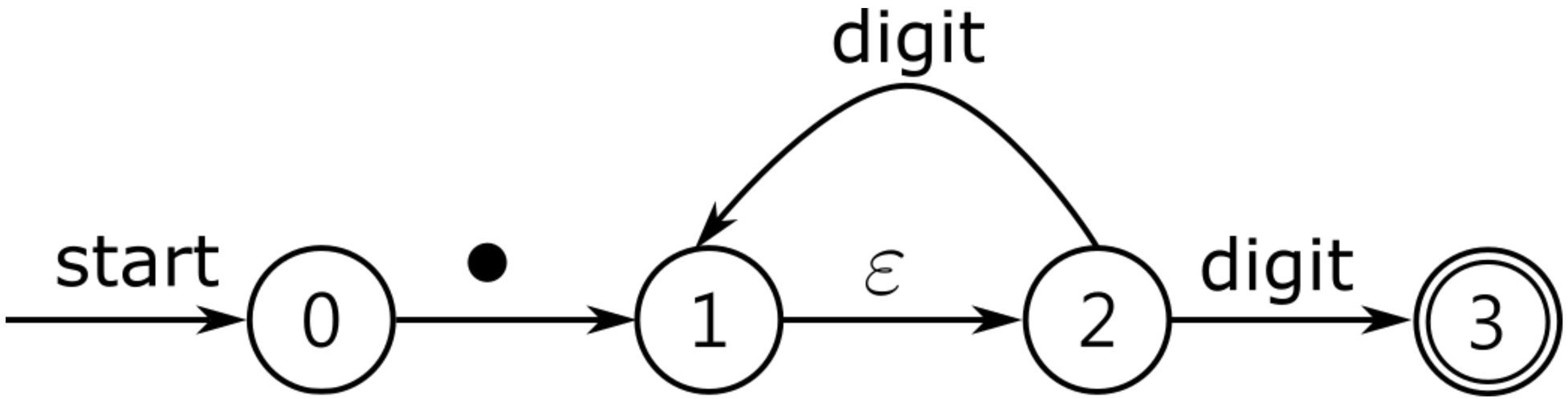
⓪ ⇒① $fraction \rightarrow . \bullet (digit)^{+}$

① ⇒② $fraction \rightarrow .(\bullet digit)^{+}$

② ⇒① $fraction \rightarrow . \bullet (digit)^{+}$

② ⇒③ $fraction \rightarrow .(digit)^{+}\bullet$ <<<<< $recognized$

fraction



optional_exponent

$$optional\_exponent \rightarrow (E(+ \mid - \mid \varepsilon)(digit)^{+}) \mid \varepsilon$$

⓪ $optional\_exponent \rightarrow \bullet(E(+ \mid - \mid \varepsilon)(digit)^{+}) \mid \varepsilon$

⓪ ⇒① $optional\_exponent \rightarrow (\bullet E(+ \mid - \mid \varepsilon)(digit)^{+}) \mid \varepsilon$
⓪ ⇒② $optional\_exponent \rightarrow (E(+ \mid - \mid \varepsilon)(digit)^{+}) \mid \bullet\varepsilon$

② ⇒③ $optional\_exponent \rightarrow (E(+ \mid - \mid \varepsilon)(digit)^{+}) \mid \varepsilon\bullet$ <<<< $recognized$

① ⇒④ $optional\_exponent \rightarrow (E \bullet (+ \mid - \mid \varepsilon)(digit)^{+}) \mid \varepsilon$

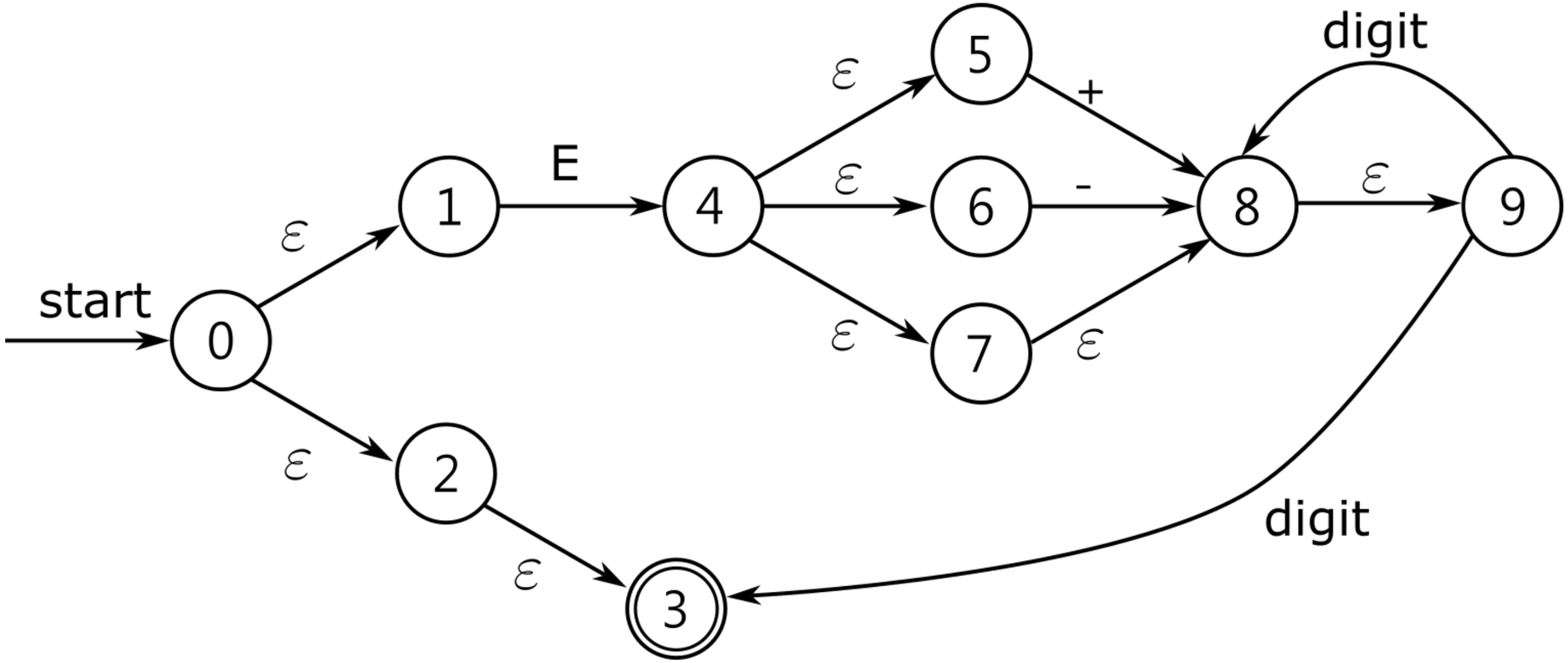
④ ⇒⑤ $optional\_exponent \rightarrow (E(\bullet + \mid - \mid \varepsilon)(digit)^{+}) \mid \varepsilon$
④ ⇒⑥ $optional\_exponent \rightarrow (E(+ \mid \bullet - \mid \varepsilon)(digit)^{+}) \mid \varepsilon$
④ ⇒⑦ $optional\_exponent \rightarrow (E(+ \mid - \mid \bullet\varepsilon)(digit)^{+}) \mid \varepsilon$

⑤⑥⑦ ⇒⑧ $optional\_exponent \rightarrow (E(+ \mid - \mid \varepsilon) \bullet (digit)^{+}) \mid \varepsilon$

⑧ ⇒⑨ $optional\_exponent \rightarrow (E(+ \mid - \mid \varepsilon)(\bullet digit)^{+}) \mid \varepsilon$

⑨ ⇒⑧ $optional\_exponent \rightarrow (E(+ \mid - \mid \varepsilon) \bullet (digit)^{+}) \mid \varepsilon$
⑨ ⇒③ $optional\_exponent \rightarrow (E(+ \mid - \mid \varepsilon)(digit)^{+}) \mid \varepsilon\bullet$ <<<< $recognized$

optional_exponent



integer

$$integer \rightarrow (digit)^{+}$$

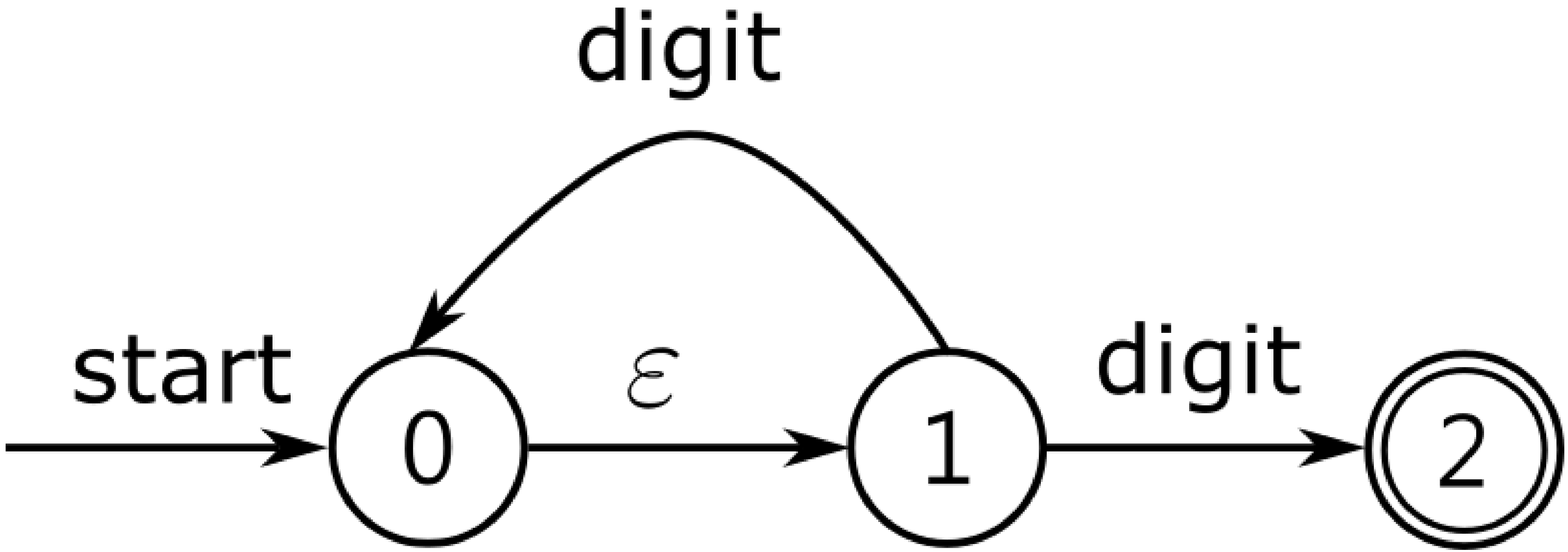
$$⓪\ integer \rightarrow \bullet(digit)^{+}$$

$$⓪ \Rightarrow ①\ integer \rightarrow (\bullet digit)^{+}$$

$$① \Rightarrow ⓪\ integer \rightarrow \bullet(digit)^{+}$$

$$① \Rightarrow ②\ integer \rightarrow (digit)^{+}\bullet <<<<< recognized$$

integer



float

$$float \rightarrow (digit)^* \ fraction \ optional\_exponent$$

⓪ $float \rightarrow \bullet(digit)^* \ fraction \ optional\_exponent$

⓪ ⇒① $float \rightarrow (\bullet digit)^* \ fraction \ optional\_exponent$

⓪ ⇒② $float \rightarrow (digit)^* \bullet \ fraction \ optional\_exponent$

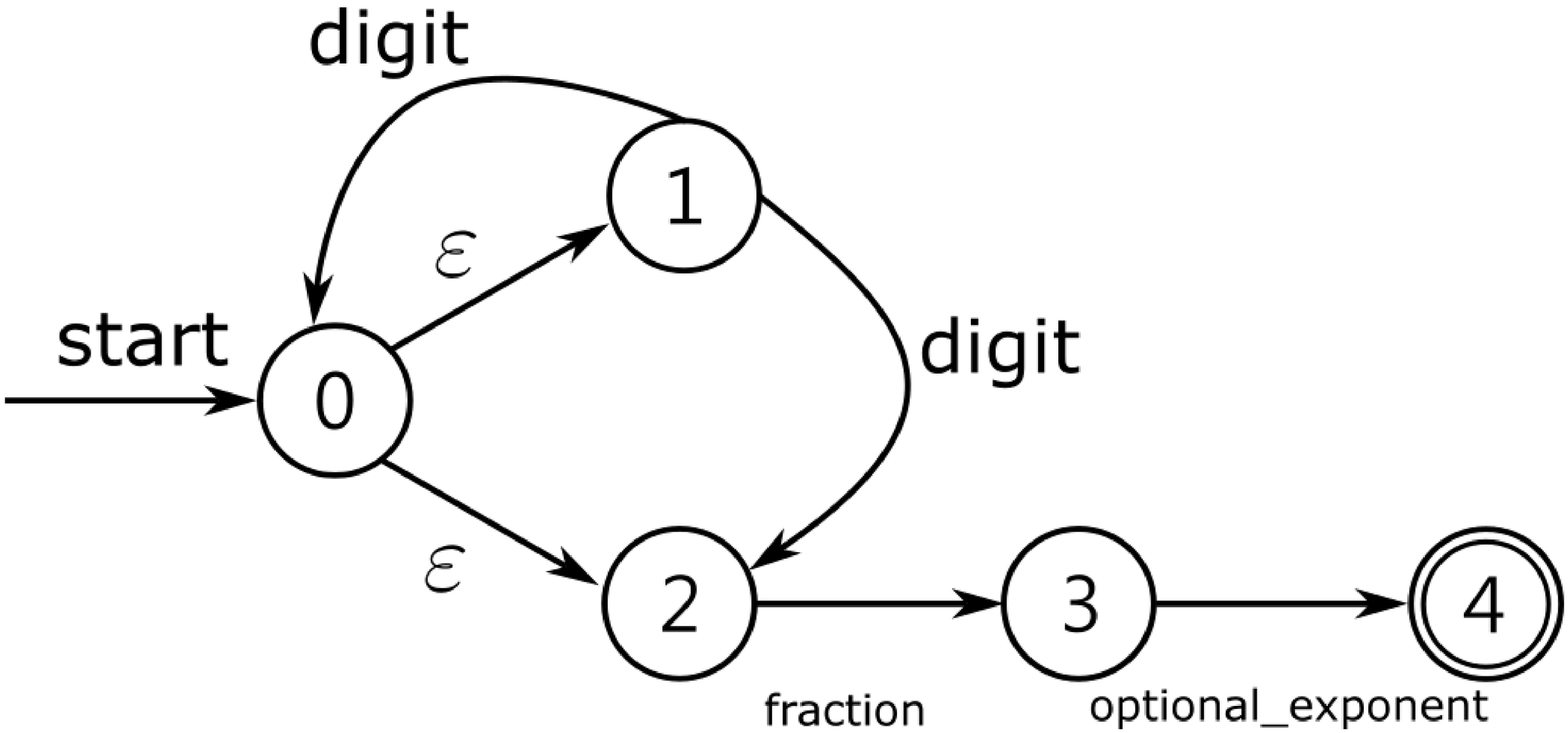
① ⇒② $float \rightarrow (digit)^* \bullet \ fraction \ optional\_exponent$

① ⇒⓪ $float \rightarrow \bullet(digit)^* \ fraction \ optional\_exponent$

② ⇒③ $float \rightarrow (digit)^* \ fraction \ \bullet optional\_exponent$

③ ⇒④ $float \rightarrow (digit)^* \ fraction \ optional\_exponent \bullet$ <<<< recognized

float



NFA -> DFA

DFA介紹

- 在任一狀態下，對於一個輸入字元，最多只能有一條對應的邊
- 不能對同一個輸入字元有兩種選擇

ε-CLOSURE介紹

- ε -closure(S)
- ε -closure(T)
- move(T, 字元)

NFA -> DFA步驟 1

Initializations:

一開始只放一個初始狀態，也就是初始項目集合，並設為 A

$$\varepsilon - closure(\{0\}) = \{0\} \Longrightarrow A$$

NFA -> DFA步驟 2

從初始狀態開始move(S, 字元)

再 ε-closure(S)

重複下去直到沒有新狀態

$A = \{0, 1, 2, 4, 7\}$

$B = \{1, 2, 3, 4, 6, 7, 8\}$

$C = \{1, 2, 4, 5, 6, 7\}$

$D = \{1, 2, 4, 5, 6, 7, 9\}$

$E = \{1, 2, 4, 5, 6, 7, 10\}$

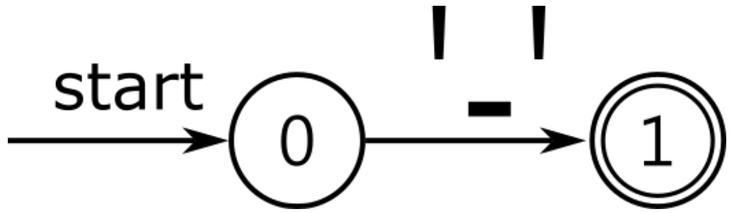
NFA -> DFA步驟3

將每個 states 化成 transition diagram

STATE	INPUT SYMBOL	
	a	*b*
A	*B*	*C*
B	*B*	*D*
C	*B*	*C*
D	*B*	*E*
E	*B*	*C*

1. underscore

1



2

$$\varepsilon - closure(\{0\}) = \{0\} \Longrightarrow A$$

$$move(A, \_) = \{1\}$$
$$\varepsilon - closure(\{1\}) = \{1\} \Longrightarrow B$$

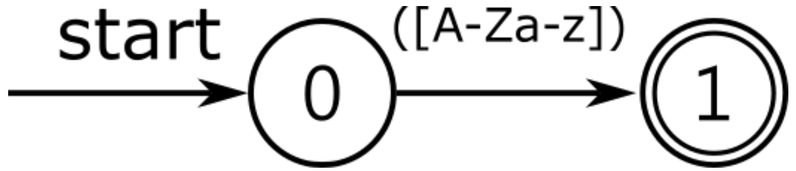
$$move(B, \_) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

3

No	State	_
0	A	B
1	B	-

2. letter

1



2

$$\varepsilon - closure(\{0\}) = \{0\} \Longrightarrow A$$

$$move(A, [A - Za - z]) = \{1\}$$
$$\varepsilon - closure(\{1\}) = \{1\} \Longrightarrow B$$

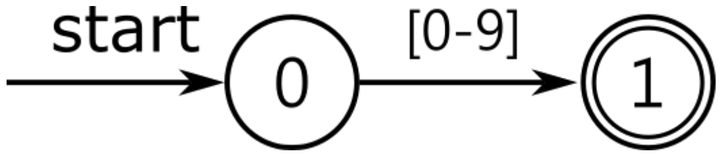
$$move(B, [A - Za - z]) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

3

No	State	[A-Za-z]
0	A	B
1	B	-

3. digit

1



2

$$\varepsilon - closure(\{0\}) = \{0\} \Longrightarrow A$$

$$move(A, [0-9]) = \{1\}$$
$$\varepsilon - closure(\{1\}) = \{1\} \Longrightarrow B$$

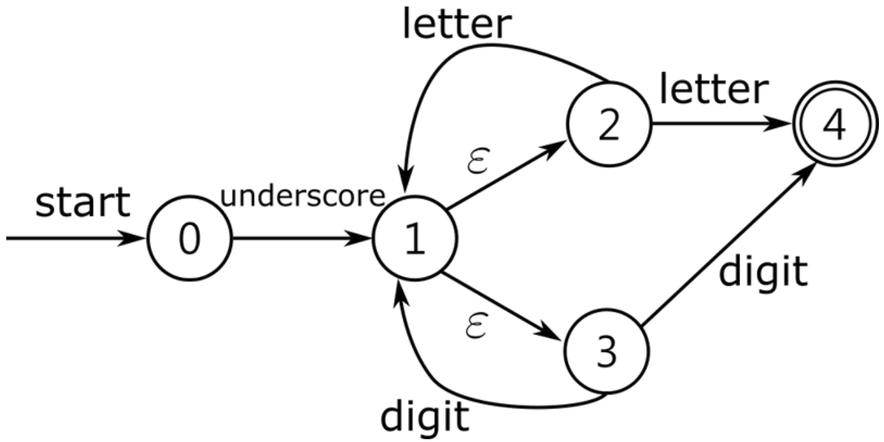
$$move(B, [0-9]) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

3

No	State	[0-9]
0	A	B
1	B	-

4. underscore_tail

1



3

No	State	underscore	letter	digit
0	A	B	-	-
1	B	-	C	C
2	C	-	C	C

2

$$\varepsilon - closure(\{0\}) = \{0\} \Longrightarrow A$$

$$move(A, underscore) = \{1\}$$
$$\varepsilon - closure(\{1\}) = \{1, 2, 3\} \Longrightarrow B$$

$$move(A, letter) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(A, digit) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, underscore) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, letter) = \{1, 4\}$$
$$\varepsilon - closure(\{1, 4\}) = \{1, 2, 3, 4\} \Longrightarrow C$$

$$move(B, digit) = \{1, 4\}$$
$$\varepsilon - closure(\{1, 4\}) = \{1, 2, 3, 4\} \Longrightarrow C$$

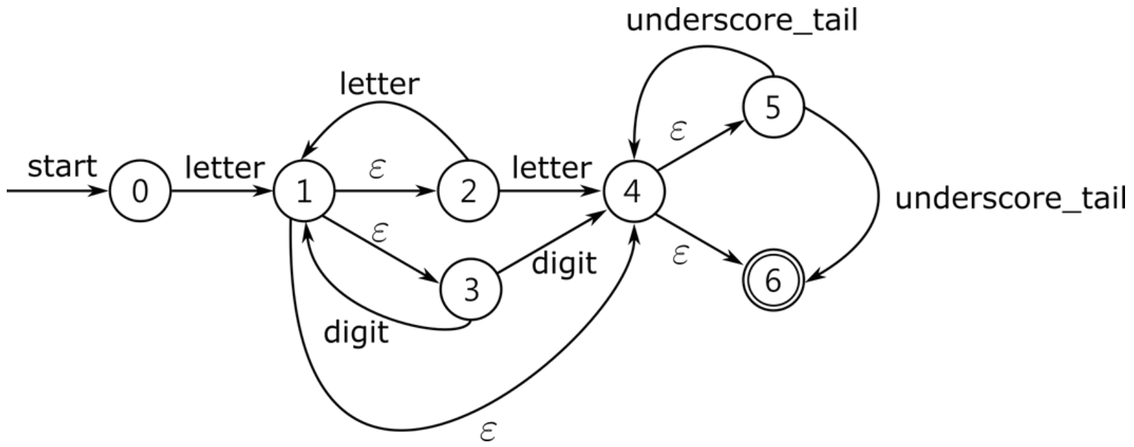
$$move(C, underscore) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C, letter) = \{1, 4\}$$
$$\varepsilon - closure(\{1, 4\}) = \{1, 2, 3, 4\} \Longrightarrow C$$

$$move(C, digit) = \{1, 4\}$$
$$\varepsilon - closure(\{1, 4\}) = \{1, 2, 3, 4\} \Longrightarrow C$$

5. id

1



2

$$\varepsilon - closure(\{0\}) = \{0\} \Longrightarrow A$$

$$move(A, letter) = \{1\}$$
$$\varepsilon - closure(\{1\}) = \{1,2,3,4,5,6\} \Longrightarrow B$$

$$move(A, digit) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(A, underscore\_tail) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, letter) = \{1,4\}$$
$$\varepsilon - closure(\{1,4\}) = \{1,2,3,4,5,6\} \Longrightarrow B$$

$$move(B, digit) = \{1,4\}$$
$$\varepsilon - closure(\{1,4\}) = \{1,2,3,4,5,6\} \Longrightarrow B$$

$$move(B, underscore\_tail) = \{4,6\}$$
$$\varepsilon - closure(\{4,6\}) = \{4,5,6\} \Longrightarrow C$$

$$move(C, letter) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

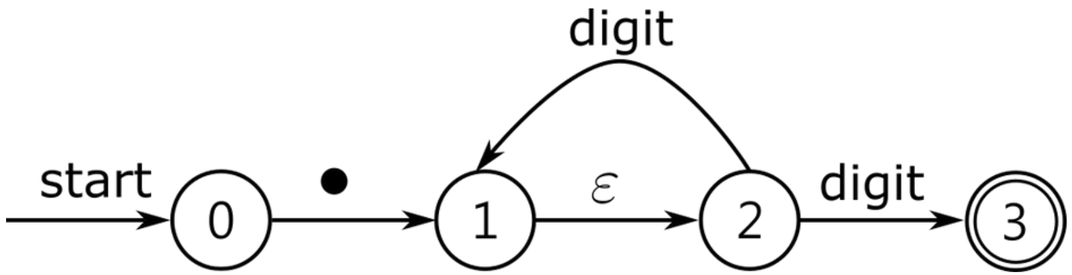
$$move(C, digit) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C, underscore\_tail) = \{4,6\}$$
$$\varepsilon - closure(\{4,6\}) = \{4,5,6\} \Longrightarrow C$$

3

No	State	letter	digit	underscore_tail
0	A	B	-	-
1	B	B	B	C
2	C	-	-	C

6. fraction



No	State	letter	digit	underscore_tail
0	A	B	-	-
1	B	B	B	C
2	C	-	-	C

$$\varepsilon - closure(\{0\}) = \{0\} \Longrightarrow A$$

$$move(A, \cdot) = \{1\}$$
$$\varepsilon - closure(\{1\}) = \{1, 2\} \Longrightarrow B$$

$$move(A, digit) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

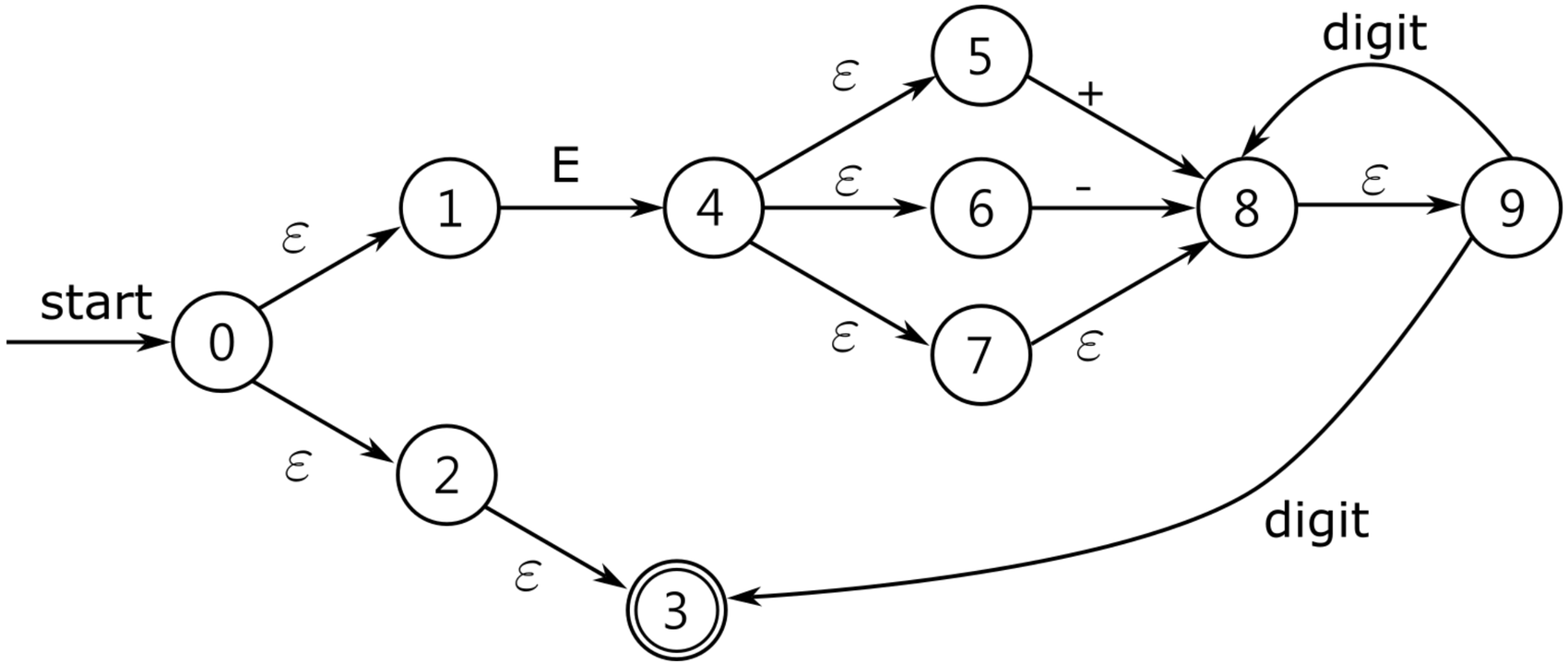
$$move(B, \cdot) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, digit) = \{1, 3\}$$
$$\varepsilon - closure(\{1, 3\}) = \{1, 2, 3\} \Longrightarrow C$$

$$move(C, \cdot) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C, digit) = \{1, 3\}$$
$$\varepsilon - closure(\{1, 3\}) = \{1, 2, 3\} \Longrightarrow C$$

7. optional_exponent



2

$$\varepsilon - closure(\{0\}) = \{0,1,2,3\} \Longrightarrow A$$

$$move(A,E) = \{4\}$$
$$\varepsilon - closure(\{4\}) = \{4,5,6,7,8,9\} \Longrightarrow B$$

$$move(A,+) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(A,-) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(A,digit) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$
$$move(B,E) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B,+) = \{8\}$$
$$\varepsilon - closure(\{8\}) = \{8,9\} \Longrightarrow C$$

$$move(B,-) = \{8\}$$
$$\varepsilon - closure(\{8\}) = \{8,9\} \Longrightarrow C$$

$$move(B,digit)\{3,8\}$$
$$\varepsilon - closure(\{3,8\}) = \{3,8,9\} \Longrightarrow D$$

$$move(C,E) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C,+) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C,-) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C,digit)\{3,8\}$$
$$\varepsilon - closure(\{3,8\}) = \{3,8,9\} \Longrightarrow D$$

$$move(D,E) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(D,+) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

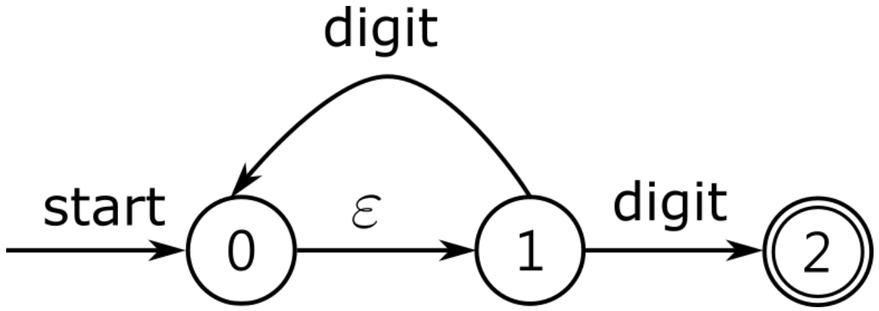
$$move(D,-) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(D,digit)\{3,8\}$$
$$\varepsilon - closure(\{3,8\}) = \{3,8,9\} \Longrightarrow D$$

No	State	E	+	-	digit
0	A	B	-	-	-
1	B	-	C	C	D
2	C	-	-	-	D
3	D	-	-	-	D

8. integer

1



2

$$\varepsilon - closure(\{0\}) = \{0,1\} \Longrightarrow A$$

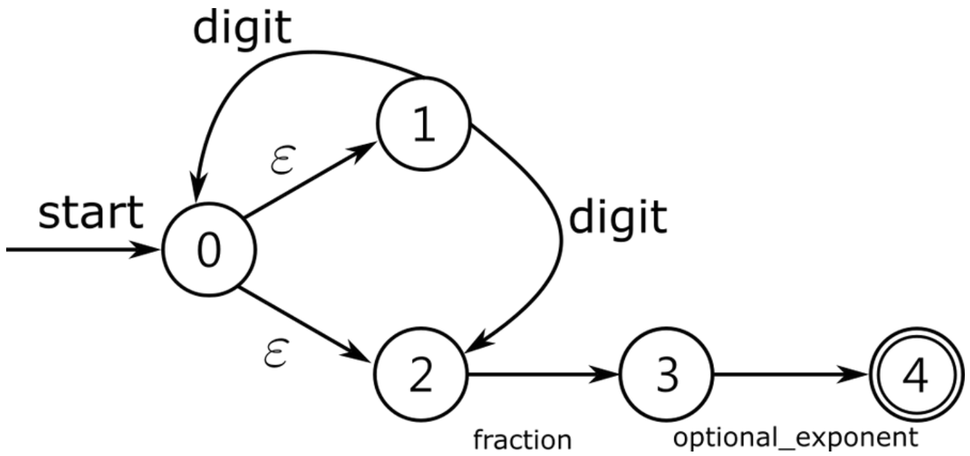
$$move(A, digit) = \{0,2\}$$
$$\varepsilon - closure(\{0,2\}) = \{0,1,2\} \Longrightarrow B$$

$$move(B, digit) = \{0,2\}$$
$$\varepsilon - closure(\{0,2\}) = \{0,1,2\} \Longrightarrow B$$

3

No	State	digit
0	A	B
1	B	B

9. float



No	State	digit	fraction	optional_exponent
0	A	A	B	-
1	B	-	-	C
2	C	-	-	-

$$\varepsilon - closure(\{0\}) = \{0,1,2\} \Longrightarrow A$$

$$move(A, digit) = \{0,2\}$$
$$\varepsilon - closure(\{0,2\}) = \{0,1,2\} \Longrightarrow A$$

$$move(A, fraction) = \{3\}$$
$$\varepsilon - closure(\{3\}) = \{3\} \Longrightarrow B$$

$$move(A, optional\_exponent) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, digit) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, fraction) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, optional\_exponent) = \{4\}$$
$$\varepsilon - closure(\{4\}) = \{4\} \Longrightarrow C$$

$$move(C, digit) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C, fraction) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C, optional\_exponent) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

DFA + DFA -> NFA -> DFA

ID

id DFA transition table

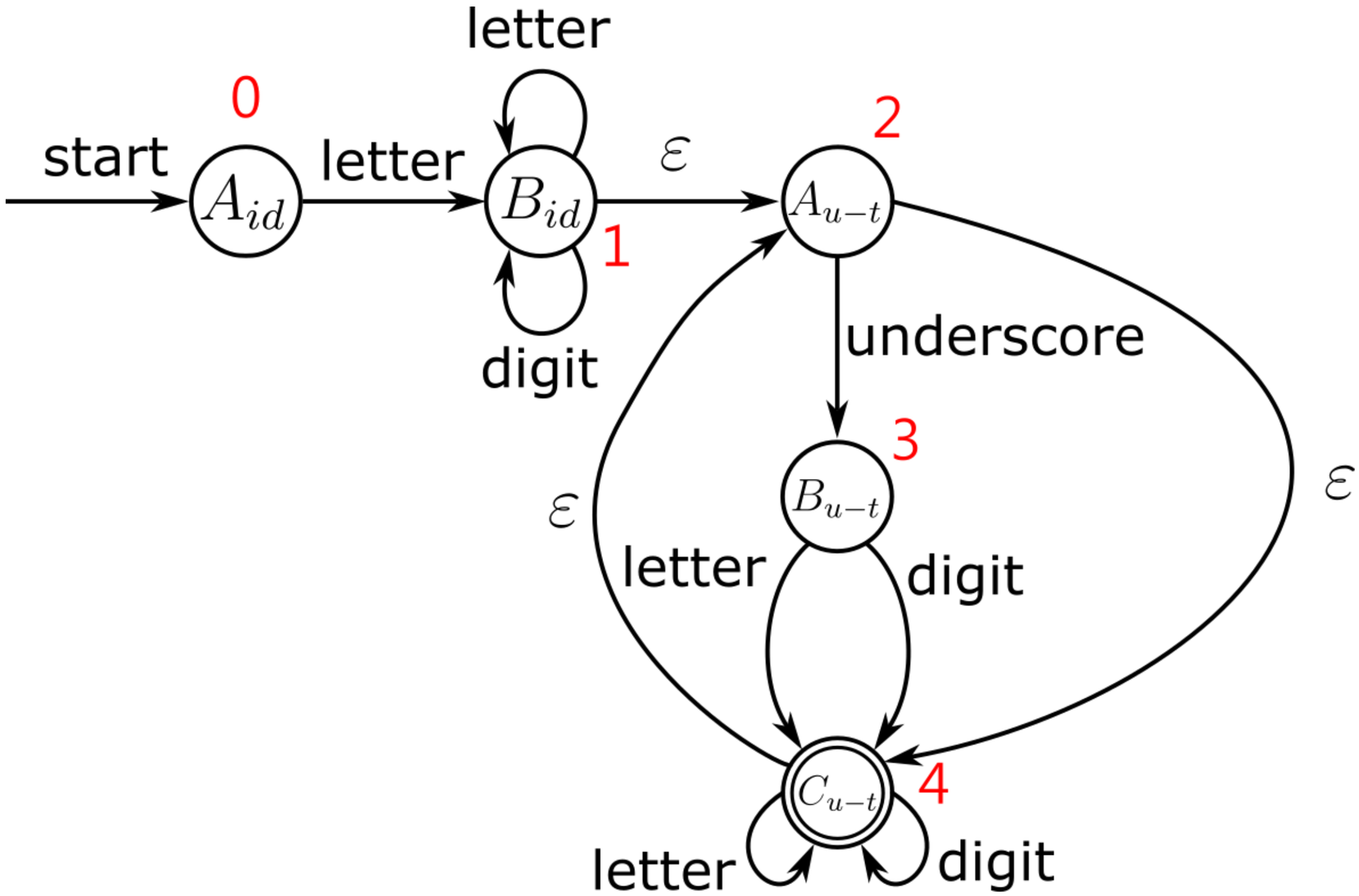
No	State	letter	digit	underscore_tail
0	A	B	-	-
1	B	B	B	C
2	C	-	-	C

ID

underscore_tail DFA transition table

No	State	underscore	letter	digit
0	A	B	-	-
1	B	-	C	C
2	C	-	C	C

NEW ID NFA


letter
0
start
letter
A_{id}
B_{id}
1
ε
2
A_{u-t}
digit
underscore
3
B_{u-t}
ε
ε
letter
digit
C_{u-t}
4
letter
digit

$$\varepsilon - closure(\{0\}) = \{0\} \Longrightarrow A$$

$$move(A, letter) = \{1\}$$
$$\varepsilon - closure(\{1\}) = \{1, 2, 4\} \Longrightarrow B$$

$$move(A, digit) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(A, underscore) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, letter) = \{1, 4\}$$
$$\varepsilon - closure(\{1, 4\}) = \{1, 2, 4\} \Longrightarrow B$$

$$move(B, digit) = \{1, 4\}$$
$$\varepsilon - closure(\{1, 4\}) = \{1, 2, 4\} \Longrightarrow B$$

$$move(B, underscore) = \{3\}$$
$$\varepsilon - closure(\{3\}) = \{3\} \Longrightarrow C$$

$$move(C, letter) = \{4\}$$
$$\varepsilon - closure(\{4\}) = \{2, 4\} \Longrightarrow D$$

$$move(C, digit) = \{4\}$$
$$\varepsilon - closure(\{4\}) = \{2, 4\} \Longrightarrow D$$

$$move(C, underscore) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(D, letter) = \{4\}$$
$$\varepsilon - closure(\{4\}) = \{2, 4\} \Longrightarrow D$$

$$move(D, digit) = \{4\}$$
$$\varepsilon - closure(\{4\}) = \{2, 4\} \Longrightarrow D$$

$$move(D, underscore) = \{3\}$$
$$\varepsilon - closure(\{3\}) = \{3\} \Longrightarrow C$$

ID TRANSITION TABLE

No	State	letter	digit	underscore
0	A	B	-	-
1	B	B	B	C
2	C	D	D	-
3	D	D	D	C

FLOAT

float DFA transition table

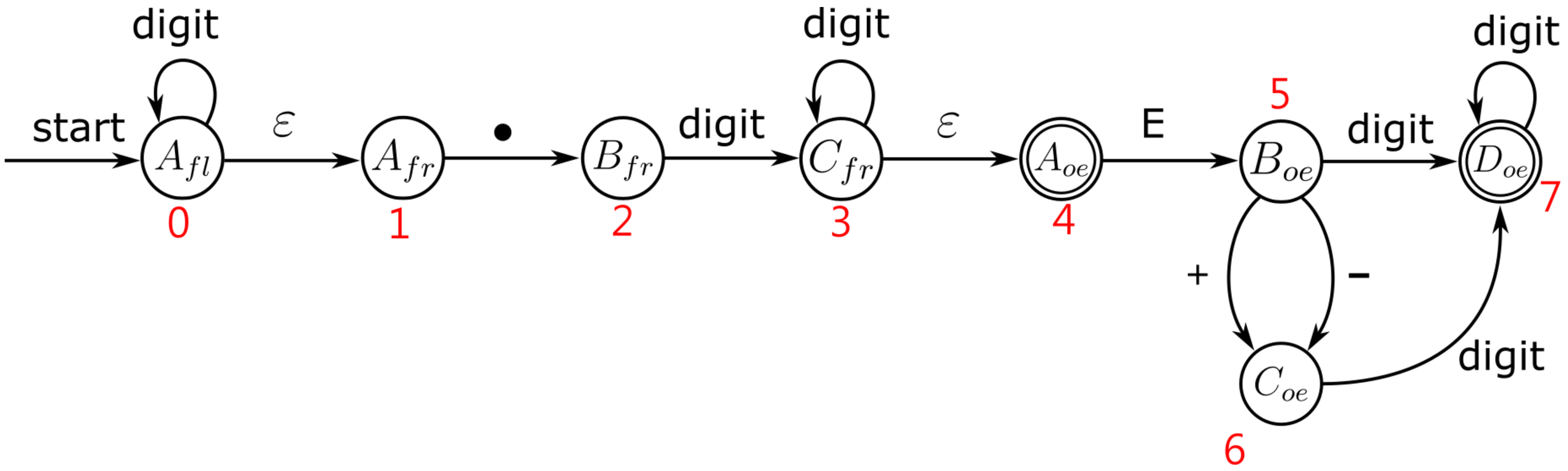
No	State	digit	fraction	optional_exponent
0	A	A	B	-
1	B	-	-	C
2	C	-	-	-

FLOAT

optional_exponent DFA transition table

No	State	E	+	-	digit
0	A	B	-	-	-
1	B	-	C	C	D
2	C	-	-	-	D
3	D	-	-	-	D

NEW FLOAT NFA



$$\varepsilon - closure(\{0\}) = \{0, 1\} \Longrightarrow A$$

$$move(A, digit) = \{0\}$$
$$\varepsilon - closure(\{0\}) = \{0, 1\} \Longrightarrow A$$

$$move(A, \cdot) = \{2\}$$
$$\varepsilon - closure(\{2\}) = \{2\} \Longrightarrow B$$

$$move(A, E) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(A, +) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(A, -) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, digit) = \{3\}$$
$$\varepsilon - closure(\{3\}) = \{3, 4\} \Longrightarrow C$$

$$move(B, \cdot) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, E) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, +) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(B, -) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C, digit) = \{3\}$$
$$\varepsilon - closure(\{3\}) = \{3, 4\} \Longrightarrow C$$

$$move(C, \cdot) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C, E) = \{5\}$$
$$\varepsilon - closure(\{5\}) = \{5\} \Longrightarrow D$$

$$move(C, +) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(C, -) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(D, digit) = \{7\}$$
$$\varepsilon - closure(\{7\}) = \{7\} \Longrightarrow E$$

$$move(D, \cdot) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(D, E) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(D, +) = \{6\}$$
$$\varepsilon - closure(\{6\}) = \{6\} \Longrightarrow F$$

$$move(D, -) = \{6\}$$
$$\varepsilon - closure(\{6\}) = \{6\} \Longrightarrow F$$

$$move(E, digit) = \{7\}$$
$$\varepsilon - closure(\{7\}) = \{7\} \Longrightarrow E$$

$$move(E, \cdot) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(E, E) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(E, +) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(E, -) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(F, digit) = \{7\}$$
$$\varepsilon - closure(\{7\}) = \{7\} \Longrightarrow E$$

$$move(F, \cdot) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(F, E) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(F, +) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

$$move(F, -) = \phi$$
$$\varepsilon - closure(\phi) = \phi$$

FLOAT TRANSITION TABLE

No	State	.	digit	E	+	-
0	A	B	A	-	-	-
1	B	-	C	-	-	-
2	C	-	C	D	-	-
3	D	-	E	-	F	F
4	E	-	E	-	-	-
5	F	-	E	-	-	-

程式

讀取字元

```
def char_type(ch):
    ascii_val = ord(ch)  # 取得 ASCII 編碼

    # 判斷是否為字母 (A-Z, a-z)
    if (65 <= ascii_val <= 90) or (97 <= ascii_val <= 122):
        return 'letter'

    # 判斷是否為數字 (0-9)
    elif 48 <= ascii_val <= 57:
        return 'digit'

    # 判斷是否為底線 (_)
    elif ascii_val == 95:
        return 'underscore'

    # 判斷是否為空白字元 (空格、tab、換行、Enter)
    elif ascii_val in (9, 10, 13, 32):
        return 'layout'

    else:
        return ch  # 其他字元
```

```
# 跳過 layout
if t == 'layout':
        i += 1
        continue
```

Separators and Brackets: the language uses ';', '(', ')', '[', ']', '{', '}' as the separators or brackets.
Binary Operators: the language uses the following operators +, -, *, /, <=, >=, <>, <, >, =, ==.
Keywords: the language uses the following keywords int, float, bool, void, while, if, else, for, return.

```
# 比較簡單的定義，只會是裡面的其中一個

# Separators
separators = {';'}

# Brackets
brackets = {'(', ')', '[', ']', '{', '}'}

# Binary Operators
binary_Operators = ['<=', '>=', '<>', '==', '+', '-', '*', '/', '<', '>', '=']

# Keywords
keywords = {'int', 'float', 'bool', 'void',
            'while', 'if', 'else', 'for', 'return'}
```

```
if dfa is dfa_comment:
            if state == 0:
                if c == '/':
                    state = 1
                else:
                    break
            elif state == 1:
                if c == '*':
                    state = 2
                else:
                    break
            elif state == 2:
                if c == '*':
                    state = 3
                else:
                    # 只要不是 '*', 留在狀態 2
                    state = 2
            elif state == 3:
                if c == '/':
                    state = 4
                elif c == '*':
                    state = 3
                else:
                    state = 2
            else:
                break

            if state in accept_states:
                last_accept = state
                last_accept_pos = i
            i += 1
            continue
```

ID TRANSITION TABLE

No	State	letter	digit	underscore
0	A	B	-	-
1	B	B	B	C
2	C	D	D	-
3	D	D	D	C

```
# id DFA
dfa_id = {
    0: {'letter': 1},
    1: {'letter': 1, 'digit': 1, 'underscore': 2},
    2: {'letter': 3, 'digit': 3},
    3: {'letter': 3, 'digit': 3, 'underscore': 2},
}
```

$$underscore \rightarrow \_$$
$$letter \rightarrow A|B...|Z|a|b...|z$$
$$digit \rightarrow 0|1|2...|9$$
$$underscore\_tail \rightarrow underscore\ (letter|digit)^{+}$$
$$id \rightarrow letter\ (letter|digit)^{*}\ underscore\_tail^{*}$$
$$fraction \rightarrow .\ digit^{+}$$
$$optional\_exponent \rightarrow (E\ (+|-|\varepsilon)\ digit^{+})|\varepsilon$$
$$integer \rightarrow\ digit^{+}$$
$$float \rightarrow digit^{*}\ fraction\ optional\_exponent$$

合法的結束狀態

```
id_accept_states = {1, 3}
```

```python
# float DFA
dfa_float = {
    0: {'digit': 0, '.': 1},
    1: {'digit': 2},
    2: {'digit': 2, 'E': 3},
    3: {'digit': 5, '+': 4, '-': 4},
    4: {'digit': 5},
    5: {'digit': 5},
}
float_accept_states = {2, 5}
```

```
# integer DFA
dfa_int = {
      0: {'digit': 1},
      1: {'digit': 1}
}
int_accept_states = {1}
```

DEMO

```
with open("input.txt", "r", encoding="utf-8") as file:
    code = file.read()
```

正常情況：輸入

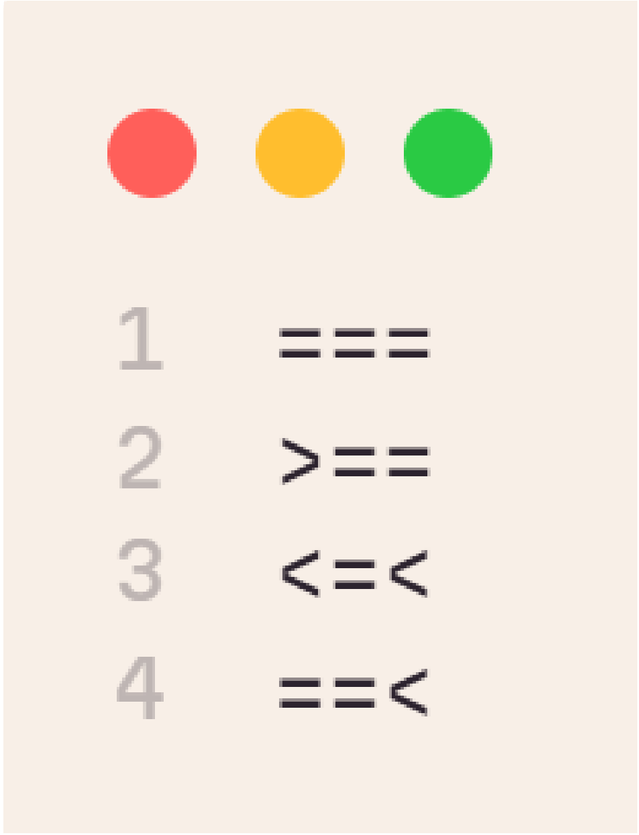
```
1  float x = 3.14E-2;
2  if (x <= 10) {
3  /* This is a comment */
4  return x + 5;
5  }
```

正常情況：輸出

```
Final  base
❯ python index.py
Keyword                 : float
id                      : x
Binary Operators        : =
float                   : 3.14E-2
Separators              : ;
Keyword                 : if
Brackets                : (
id                      : x
Binary Operators        : <=
integer                 : 10
Brackets                : )
Brackets                : {
Comment                 : /* This is a comment */
Keyword                 : return
id                      : x
Binary Operators        : +
integer                 : 5
Separators              : ;
Brackets                : }
```

特殊情況：輸入

- 最長字串匹配



特殊情況：輸出

- 最長字串匹配

```
Final  base
❯ python index.py
Binary Operators     :  ==
Binary Operators     :  =
Binary Operators     :  >=
Binary Operators     :  =
Binary Operators     :  <=
Binary Operators     :  <
Binary Operators     :  ==
Binary Operators     :  <
```

特殊情況：輸入

- 錯誤輸出

```
1  if (x <= 10) {
2  /* This is a comment */
3  return x + 5; @
4  }
```

特殊情況：輸出

- 錯誤輸出

```
Final  base
● ❯ python index.py
Keyword                 : if
Brackets                : (
id                      : x
Binary Operators        : <=
integer                 : 10
Brackets                : )
Brackets                : {
Comment                 : /* This is a comment */
Keyword                 : return
id                      : x
Binary Operators        : +
integer                 : 5
Separators              : ;

Lexer Error: Unknown token starting at position 61: @
```

特殊情況：輸入

- Comments

```
/* comments 一段奇怪的符號 @$$@%^@#$!# */
```

特殊情況：輸出

- Comments

```
Final  base
● ❯ python index.py
Comments                : /* comments 一段奇怪的符號 @$$@%^@#$!# */
```

特殊情況：輸入

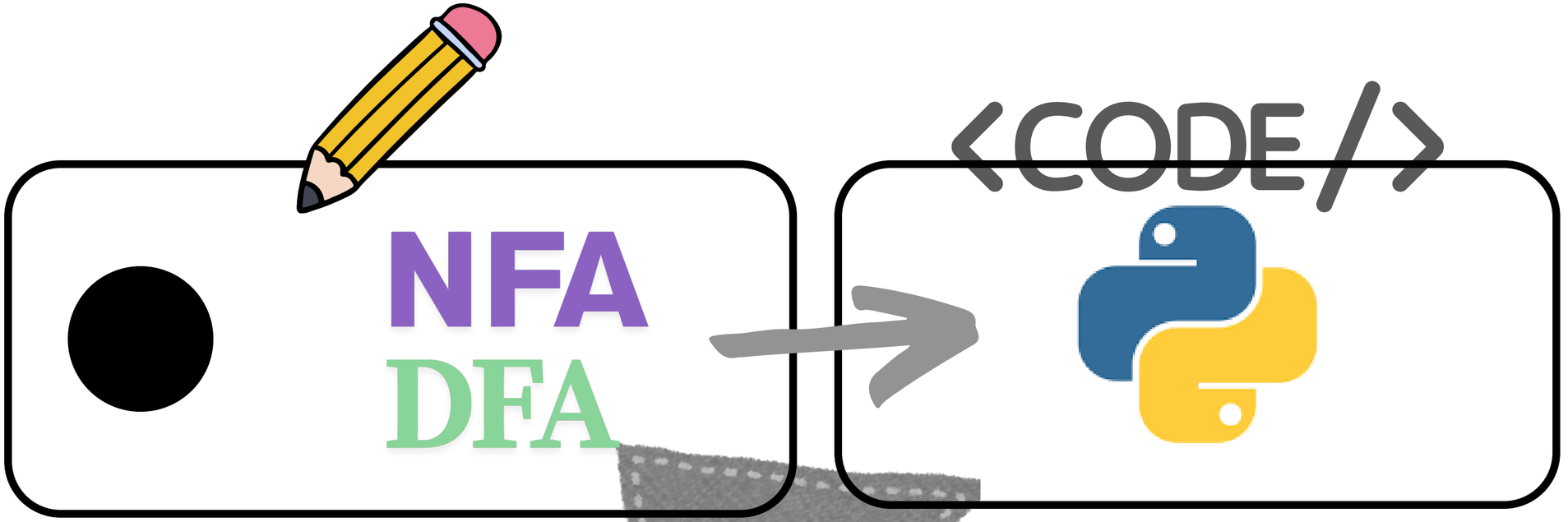
- Comments

特殊情況：輸出

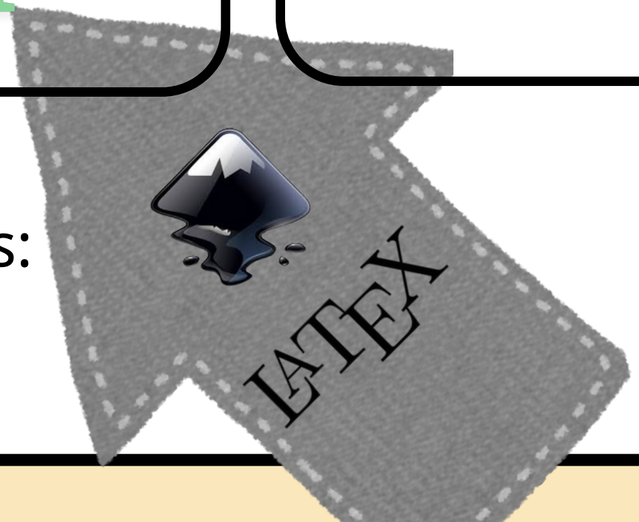
- Comments

```
Final  base
● ❯ python index.py
Binary Operators     : /
Binary Operators     : *
id                   : comments
Binary Operators     : /
```

結論



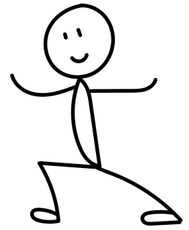
tools:



分工

顏琦恩
22%

- Regular Expression --> NFA
- NFA -> DFA
- 簡報製作
- 影片剪輯

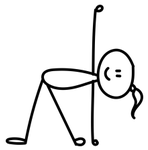


宋冠穎
25%

- Regular Expression -> NFA
- NFA -> DFA
- Demo
- 簡報製作
- 使用 LaTex 將 Dotted Item 手寫過程整理成數位檔案

王嘉汶
18%

- 繪圖
- Demo
- 簡報製作

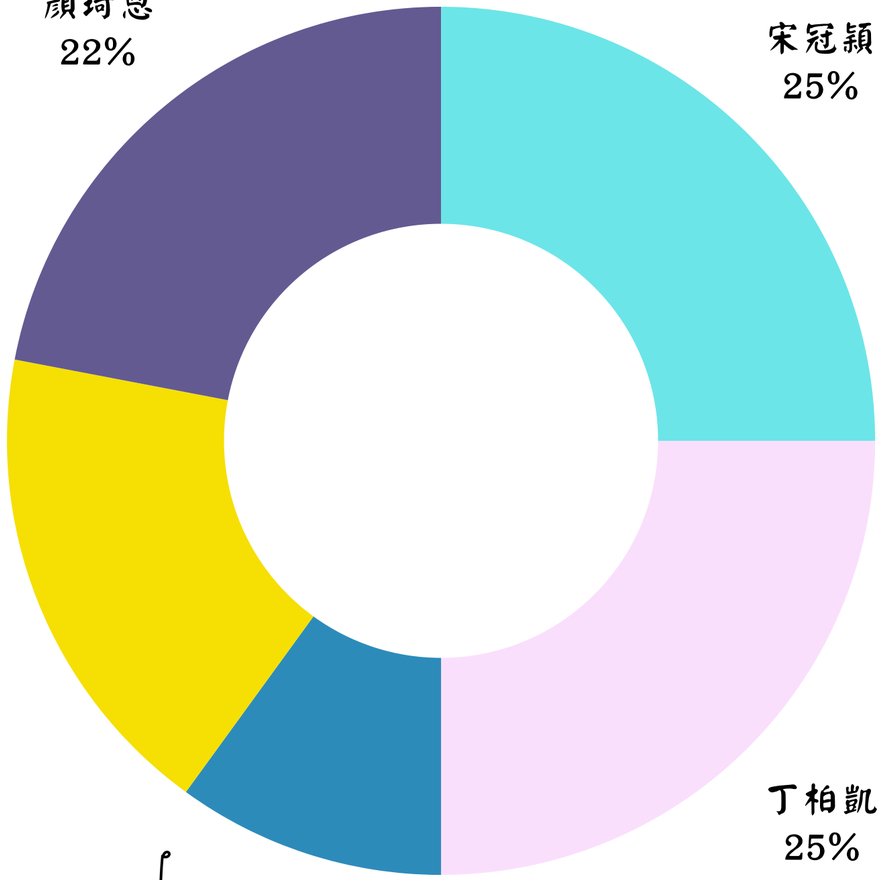


詹佳瑜
10%

- 想測資
- 檢查輸入 & 輸出

丁柏凱
25%

- NFA -> DFA
- Python Program
- Demo
- 簡報製作
- 使用 LaTex 將 Dotted Item 手寫過程整理成數位檔案



參考資料

- 王智弘老師上課　PPT

THANK YOU