

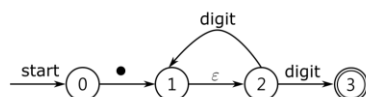
我們作業的流程是：

Regular Expression --> NFA (Dotted Item Method) --> DFA
--> Python Program

我主要負責處理從 NFA 圖形轉成 DFA 的 Transition table，以及將 DFA 的轉換應用到 Python Program 這兩部分。

一、將 NFA 圖形轉成 DFA 的 Transition table

其他組員使用 Dotted Item 的方法，把正規表示式畫成 NFA 圖形，為了要放入報告中，我們使用了 inkscape 這個軟體，將手繪的 NFA 圖形轉成電子檔案，如下圖所示：



不過只有我會用這個軟體，所以這部分是由我先教其他組員怎麼使用 inkscape，交給他去畫圖，完成後我再進行檢查和修正。

我依據 NFA 圖形，先用手寫的方式，計算出每一個的 DFA Transition table:

underscore - tail

$$\xi(0) = 0 \Rightarrow A$$

$$(A, u) = 1, \xi(1) = 1, 2, 3 \Rightarrow B$$

$$(B, l) = 1, 4, \xi(1, 4) = 1, 2, 3, 4 \Rightarrow C$$

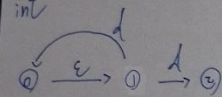
$$(B, d) = 1, 4 \Rightarrow C$$

$$(C, l) = 1, 4 \Rightarrow C$$

$$(C, d) \Rightarrow C$$

	u	l	d
A	B		
B		C	C
C		C	C

int

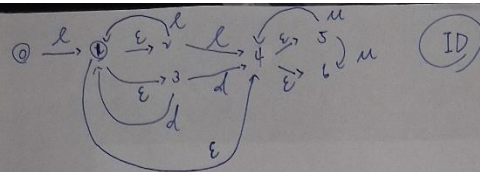


$$\xi(0) = 0, 1 \Rightarrow A$$

$$(A, d) = 0, 1 \Rightarrow \xi(0, 1) = 0, 1, 1 \Rightarrow B$$

$$(B, d) = 0, 1 \Rightarrow B$$

	d
A	B
B	B



	e	d	u
A	B		
B	B	B	C
C			C

$$\xi(0) = 0, \Rightarrow A$$

$$(A, l) = 1, \xi(1) = 1, 2, 3, 4, 5, 6 \Rightarrow B$$

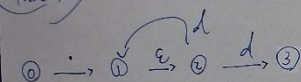
$$(B, l) = 1, 4, \xi(1, 4) = 1, 2, 3, 4, 5, 6 \Rightarrow B$$

$$(B, d) = 1, 4, \Rightarrow B$$

$$(B, u) = 4, 6, \xi(4, 6) = 4, 5, 6 \Rightarrow C$$

$$(C, u) = 4, 6 \Rightarrow C$$

fraction



$$\xi(0) = 0 \Rightarrow A$$

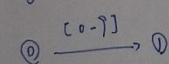
$$\text{move}(A, \cdot) = 1, \xi(1) = 1, 2 \Rightarrow B$$

$$\text{move}(B, d) = 1, 3, \xi(1, 3) = 1, 2, 3 \Rightarrow C$$

$$\text{move}(C, d) = 1, 3 \Rightarrow C$$

	.	d
A	B	-
B	-	C
C	-	C

digit



$$\xi(0) = 0 \Rightarrow A$$

$$(A, [0-9]) = 1, \xi(1) = 1 \Rightarrow B$$

	[0-9]
A	B
B	

$$\xi(1, 1) = 1, \xi(1, 1) = 1, 2 \Rightarrow B$$

Optional

$$\epsilon(0) = 0, 1, 2, 3 \Rightarrow A$$

$$(A, E) = 4 \Rightarrow \epsilon(4) = 4, 5, 6, 7, 8, 9 \Rightarrow B$$

$$(B, +) \Rightarrow 8, \epsilon(8) = 8, 9 \Rightarrow C$$

$$(B, -) \Rightarrow C$$

$$(B, d) \Rightarrow 8, 3, \epsilon(3, 8) = 3, 8, 9 \Rightarrow D$$

$$(D, d) \Rightarrow 8, 3 \Rightarrow D$$

$$(C, d) \Rightarrow 3, 8 \Rightarrow D$$

	E	+	-	d
✓ A	B			
B		C	C	D
C				D
✓ D				D

float

$$\epsilon(0) = 0, 1, 2 \Rightarrow A$$

$$(A, d) = 0, 2 \Rightarrow A, \epsilon(2, 0) = 0, 1, 2 \Rightarrow A$$

$$(A, f) = 3, \epsilon(3) = 3 \Rightarrow B$$

$$(B, 0) = 4, \epsilon(4) = 4 \Rightarrow C$$

	d	f	0
A	A	B	
B			C
✓ C			

underscore

$$\textcircled{2} \rightarrow \textcircled{1}$$

$$\epsilon(0) = 0 \Rightarrow A$$

$$\text{move}(A, -) = 1 \Rightarrow B$$

A	B
✓ B	

letter

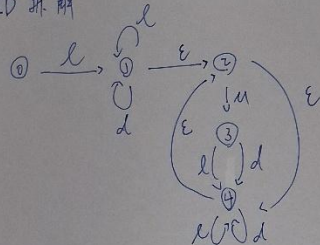
$$\textcircled{2} [A-Z a-z] \rightarrow \textcircled{1}$$

$$\epsilon(0) = 0 \Rightarrow A$$

$$(A, [A-Z a-z]) \rightarrow 1 \Rightarrow B$$

	[A-Z a-z]
A	B
✓ B	

ID 拆解



$$\epsilon(0) = 0 \Rightarrow A$$

$$(A, l) = 1, \epsilon(1) = 1, 2, 4 \Rightarrow B$$

$$(B, e) = 1, 4 \Rightarrow \epsilon(1, 4) = 1, 2, 4 \Rightarrow B$$

$$(B, d) = 1, 4 \Rightarrow B$$

$$(B, u) = 3, \epsilon(3) = 3 \Rightarrow C$$

$$(C, e) = 4, \epsilon(4) = 2, 4 \Rightarrow D$$

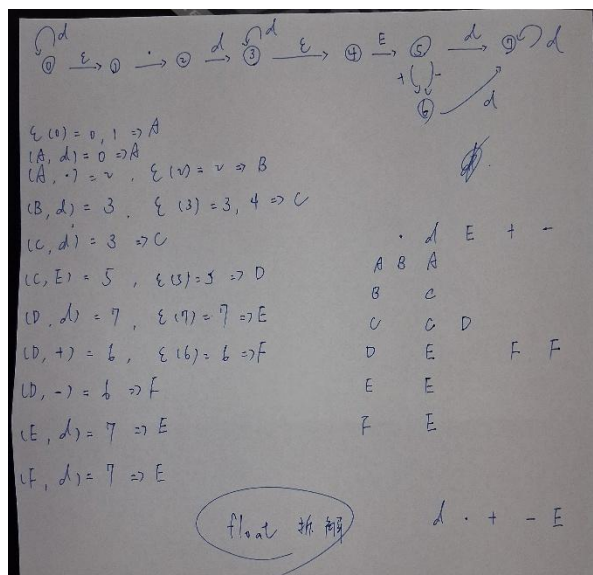
$$(C, d) = 4 \Rightarrow D$$

$$(D, e) = 4 \Rightarrow D$$

$$(D, d) = 4 \Rightarrow D$$

$$(D, u) = 3 \Rightarrow C$$

	l	d	u
A	B		
✓ B	B	B	C
C	D	D	
✓ D	D	D	C



完成之後，也因為是要放入報告中，所以需要轉成電子文件，

我畫的 NFA --> DFA 就由我自己來完成。

(Regular Expression --> NFA 是由其他組員完成)

我使用 Markdown + LaTeX 來完成，像是其中一個部分：

```

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

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

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

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

```

$$move(C, underscore) = \phi$$

$$\varepsilon - closure(\phi) = \phi$$

$$move(C, letter) = \{1, 4\}$$

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

$$move(C, digit) = \{1, 4\}$$

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

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

Markdown

PDF

(上傳檔案有附上完整的 .md 和 PDF 檔案)

二、將 DFA 的轉換應用到 Python Program

我們的程式是用 Python 來實作，全部都是我自己一個人寫的。(上傳檔案有附上程式)

輸入是讀取一個檔案，所以要在 index.py 同目錄下放一個 input.txt 作為輸入測資。

一開始想說，是不是全部的規則都要能夠被判斷出來，像是 underscore_tail 這個規則，如果輸入 _AErwev6，應該要被判斷成是 underscore_tail，但是後面在處理 float 的時候，有遇到像是 fraction 完全被 float 覆蓋、輸入 E10 要被判斷成 optional_exponent 還是 id 等等問題，後來才發現題目其實有要求要判別出哪幾個...

要如何將 DFA 的過程，套入到程式中呢?我是用 python 中的 dictionary。舉個例子來說明:

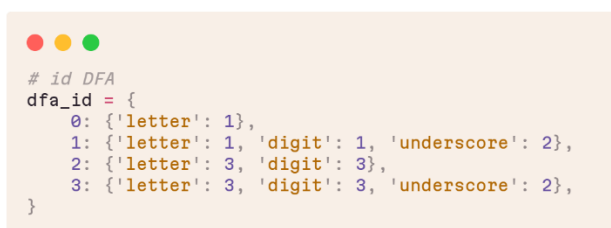
No	State	letter	digit	underscore
0	A	B	-	-
1	B	B	B	C
2	C	D	D	-
3	D	D	D	C

先把原本寫的 A~D 個狀態，標為 0~4

從原本初始狀態 A，也就是 0，如果讀了 letter，就會跑到狀態 B，也就是狀態 1，那程式中就寫上，狀態 0，讀取 letter，就切換到狀態 1。

那在狀態 B，也就是 1，如果讀取了 letter 或是 digit，還是會留在狀態 B，讀取 underscore 會轉換到狀態 C，那程式中就寫上，狀態 1，讀取 letter，切換到狀態 1，讀取 digit，切換到狀態 1，讀取 underscore，切換到狀態 2。後面依此類推。

我們就可以在 dictionary 中寫下這樣的狀態轉移：



```
# 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},
}
```

再來還需要決定一下合法的結束狀態：

狀態 A 是初始狀態，當然不能當作是結束狀態。

而狀態 B，是從 A 讀了英文字母之後，或是 B 自己讀了字元或數字，也就是這串輸入的最後是英文字母或是數字，因此狀態 B 是可以作為結束狀態的。

狀態 C，是狀態 B 或 D 讀取了底線而來，也就是這串輸入的

最後是底線，那根據規則，id 的結尾不可能是底線，所以狀態

C 不能當作結束狀態。

狀態 D，是由狀態 C 讀了英文字母或數字而來，也就是底線後

面跟了英文字母或數字，或是從狀態 D 再讀英文字母或數字，

因此這串輸入的最後面是英文字母或是數字，可以當作結束狀

態。

id 這個 DFA，可以做為結束狀態的有 B 和 D，也就是狀態 1

和 3，因此我們把合法的結束狀態記錄下來：



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

後面幾種也都是一樣的做法。

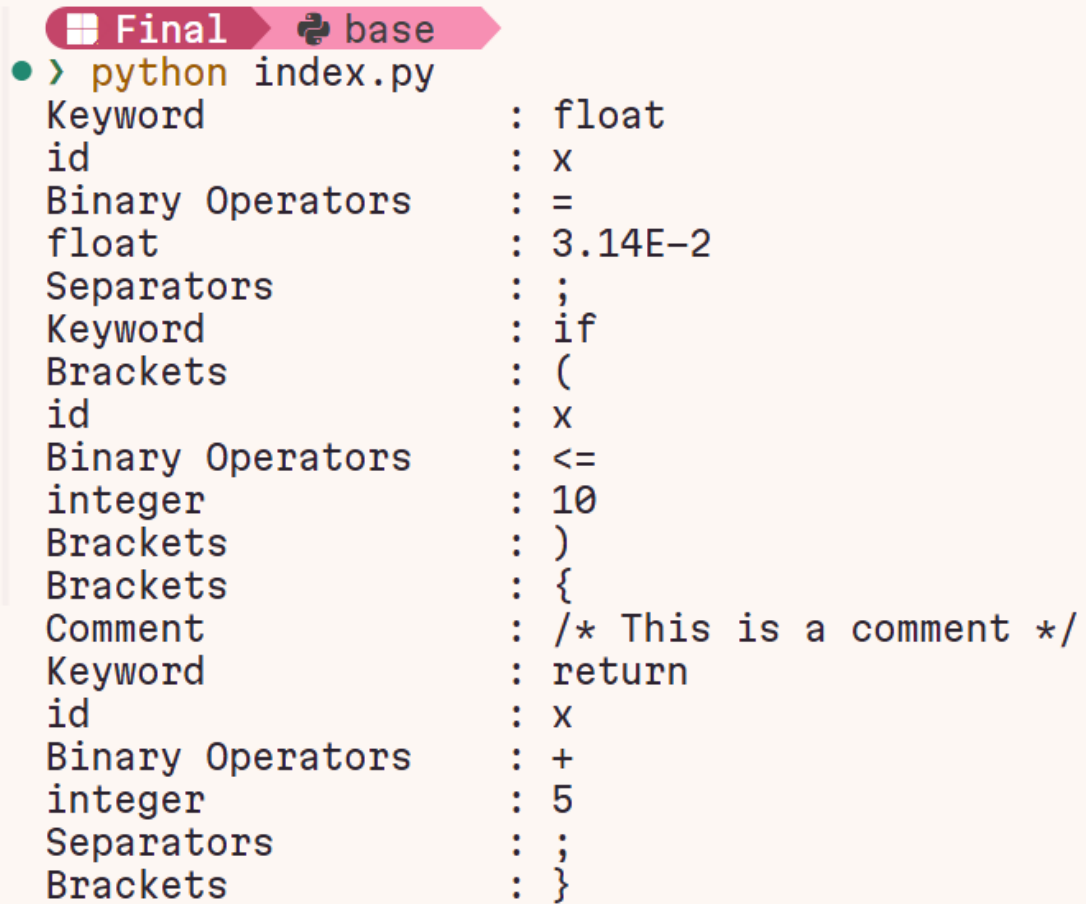
測試一下執行結果

輸入：



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

可以得到這樣的輸出：



```
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         : }
```

Token 都有被判別出來，是沒有問題的。