

- 測資結果
 - 110612008_沈昱翔_HW6(for string)

```

C:\Users\40924\Desktop\C\AI X + v
X=?
introduction
Y=?
construction
Z=?
instruction
LCS Length: 9
LCS: ntruction

-----
Process exited after 28.23 seconds with return value 0
Press any key to continue . . .

```

- 110612008_沈昱翔_HW6(for float)

```

C:\Users\40924\Desktop\C\AI X + v
X=?
1 1.4 1.5 1.7 2 2.2 2.5 2.6 2.9
Y=?
1.1 1.4 1.6 1.8 2 2.1 2.4 2.9 3
Z=?
1.3 1.4 1.7 2 2.1 2.5 2.6 2.7 2.9
LCS: 1.4 2 2.9
LCS Length: 3

-----
Process exited after 67 seconds with return value 0
Press any key to continue . . .

```

- 討論
 - 如果我們使用窮舉法，則需要考慮全部的子序列組合，因為每個子序列組合都有可能，因此如果輸入三個字串 X, Y, Z，長度分別為 m, n, o，窮舉法的時間複雜度即為 $O(2^m \times 2^n \times 2^o) = O(2^{m+n+o})$ 。也就是說，當字串長度增加或是輸入字串的數量增加，計算時間會指數上升。
 - 老師使用動態規劃，通過儲存中間的結果避免重複計算，跟窮舉法相比，這樣設計大大減少了時間複雜度。
 - 本程式先創建一個 $(m + 1) \times (n + 1) \times (o + 1)$ 的三維 vector C 和 b，並初始化為 0，之後開始推進，副程式 LCS_length 中用三個 for loop 來寫入 C 和 b，每個單元格需要 $O(1)$ 的時間來填充，也就是說，LCS_length 的時間複雜度為 $O(m \times n \times o)$ ，另外一個副程式 Print_LCS 的是一個遞迴，在 worst case，每次遞迴都只會讓 i 或 j 或 k 減少一，也就是說，Print_LCS 的時間複雜度為 $O(m + n + o)$ 。
 - 由此可知兩個副程式的時間複雜度相加即為整個程式的複雜度， $O(m \times n \times o) + O(m + n + o) = O(m \times n \times o)$ 。