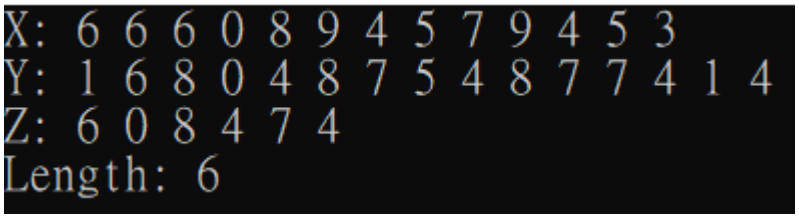
0819823 陳子祈 演算法導論 HW6

一、 測試演算法正確性

給定 X=[6,6,6,0,8,9,4,5,7,9,4,5,3], Y=[1,6,8,0,4,8,7,5,4,8,7,7,4,1,4]
請大家利用老師上課所講的得出 Longest Common Subsequence 的演算法，找出 Z=LCS(X,Y)以及印出 Longest Common SubsequenceZ 的長度。

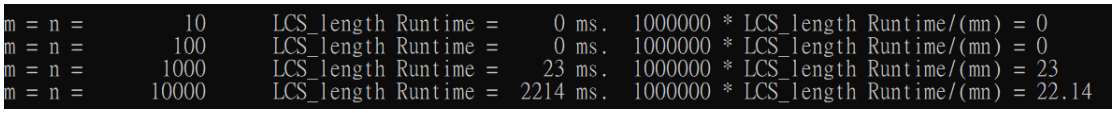


二、 時間複雜度試驗成果

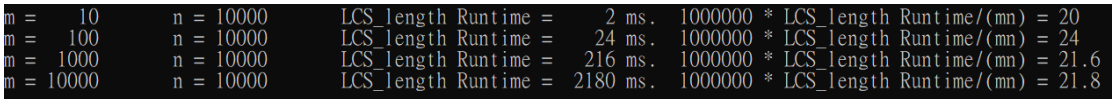
嘗試產生隨機元素為0~9不同長度的X與Y，並記錄下不同長度時所花費的時間，自行測試不同長度以驗證LCS-Length和Print-LCS時間複雜度是否和上課所講的一樣。

(一) LCS_length

根據上述基本要求，每個 input array 所有元素都是從 0~9 隨機產生的，並使用 for 迴圈設定了 m 從 10 逐次成長 10 倍到 10000，根據實測結果如下:



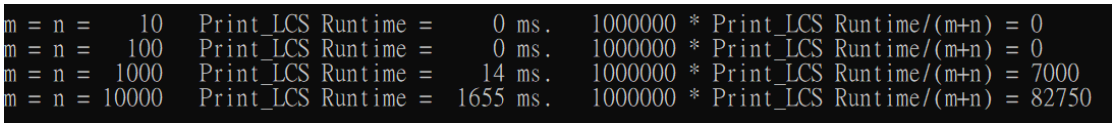
當 m、n 小於 1000 時，因為執行時間太短測不到，無法比較時間複雜度是否與上課講得一樣，因此我多測了幾項數據如下:



由上表可知，執行時間與 m*n 有接近正比的關係，若 n 為 input size，f(m,n)為此演算法執行的時間，符合Θ(f(m,n)) = m*n 的趨勢。

(二) Print_LCS

根據上述基本要求，每個 input array 所有元素都是從隨機產生的，並使用 for 迴圈設定了 m 從 10 逐次成長 10 倍到 10000，根據實測結果如下:



當 m、n 小於 1000 時，因為執行時間也會太短測不到，無法比較時間複雜度是否與上課講得一樣，因此我多測了幾項數據如下:

```
m =    10       n = 10000       Print_LCS Runtime =     0 ms.   1000000 * Print_LCS Runtime/(m+n) = 0
m =   100       n = 10000       Print_LCS Runtime =     1 ms.   1000000 * Print_LCS Runtime/(m+n) = 99.0099
m =  1000       n = 10000       Print_LCS Runtime =   142 ms.   1000000 * Print_LCS Runtime/(m+n) = 12909.1
m = 10000       n = 10000       Print_LCS Runtime =  1666 ms.   1000000 * Print_LCS Runtime/(m+n) = 83300
```

由上表可知，執行時間與 m+n 無正比的關係，若 n 為 input size，f(m,n)為此演算法執行的時間，不符合Θ(f(m,n)) = m+n 的趨勢。最大的原因是 Print_LCS 以遞迴呼叫的方式實作，function call 花的時間比 iteration 花的時間長很多，m、n 線性成長不會讓執行時間也線性成長。

三、 討論

Q: 試著解釋原先窮舉法的時間複雜度為何，以及老師所講的演算法是如何改善時間複雜度。

A: 原先窮舉法的方法應該是找出 X、Y 所有的 Subsequence，先將所有 Subsequence 比較是不是 Common Subsequence，再找出最長的 Common Subsequence。但是光 X 的所有的 Subsequence 就有 2^m 個，再兩兩比較是不是 Common Subsequence 就要花 $2^m * 2^n = 2^{m+n}$ 次比較，執行時間會很長。因此老師所講的演算法使用 Dynamic programming 的技巧，提出以下構想:

Theorem 1. Let $X = [x_1, x_2, \ldots, x_m]$ and $Y = [y_1, y_2, \ldots, y_n]$. If $Z = [z_1, z_2, \ldots, z_k]$ is a LCS of X and Y, then we have

1. If $x_m = y_n$, then $z_k = x_m = y_n$ and Z_{k-1} is an LCS of X_{m-1} and Y_{n-1}.
2. If $x_m \neq y_n$, then $z_k \neq x_m$ implies that Z is an LCS of X_{m-1} and Y.
3. If $x_m \neq y_n$, then $z_k \neq y_n$ implies that Z is an LCS of X and Y_{n-1}

當 $x_m = y_n$ 時，z_k 就是 LCS 其中的一個元素，因此可以讓 i=1~m、j=1~n 跑，當找到 $x_i = y_j$，就把這個值存到 z 裡面，因此執行時間的複雜度就降為 m*n。只是講義上寫的是先找出 LCS 的長度再找出 LCS 每個元素的值，沒有直接把那個值存到 z，不過方法大同小異，時間複雜度可以改善就好。

四、 心得

這一次作業與前幾次較時間複雜度的作業很像，不過這次要測試 Longest Common Subsequence 程式的執行時間。需要實作的函式只有 LCS 的兩個函式不多，其演算法也很簡單，因此這次作業主要就是要思考用什麼方法時做出來。我想到使用 vector 的方法實作，因為 vector 的創建與去除都比較靈活，可輕鬆創建隨機大小的 vector 拿去測試 LCS。測試結果發現使用 for loop 的執行時間與預想一樣，但是使用遞迴呼叫的執行時間比預想長很多，我覺得如果把遞迴呼叫改成 iteration 的方法，執行時間應該會跟上課的一樣。