Algorithm HW6_report

A. 程式說明

此題為 Longest Common Subsequence(LCS)，與課本不同的是此題要找的是 3 個 sequence 的 LCS，使用者輸入 3 個 sequence 的長度(此長度限制在 1~100)，程式會隨機產生 3 個輸入長度的陣列，且每個元素的值都介於 0~49，如此能夠產生更長的 Longest Common Subsequence，若要觀察更長的 LCS，可以將元素的值限制於 0~10。因為此為 3 個 sequence，所以需要用到 3D-array，而其他部分則與課本的 pseudo code 類似。

B. 程式結果

第一行需先輸入 3 個 sequence 的長度，接下來因為陣列的值是隨機產生的，所以接下來的三行為 X，Y，Z 的陣列值，而接下來的兩行分別為 Longest Common Subsequence W 以及它的長度。

```
Please input the length of three sequences (1~100) :
50 50 50
X : 33 36 27 15 43 35 36 42 49 21 12 27 40 9 13 26 40 26 22 36 11 18 17 29 32 30 12 23 17 35 29 2 22 8 19 17 43 6 11 42 29 23 21 19 34 37 4
8 24 15 20
Y : 13 26 41 30 6 23 12 20 46 31 5 25 34 27 36 5 46 29 13 7 24 45 32 45 14 17 34 14 43 0 37 8 26 28 38 34 3 1 4 49 32 10 26 18 39 12 26 36
44 39
Z : 45 20 34 28 17 1 47 2 17 42 2 6 1 30 36 41 15 39 44 19 40 29 31 17 47 21 31 25 9 27 17 6 47 3 36 15 6 33 19 24 28 21 32 29 3 19 20 18 8
 15
Longest Common Subsequence W :
27 36 29 8

Length of W : 4
```

(值限制在 0~49)

```
Please input the length of three sequences (1~100) :
50 50 50
X : 3 6 7 5 3 5 6 2 9 1 2 7 0 9 3 6 0 6 2 6 1 8 7 9 2 0 2 3 7 5 9 2 2 8 9 7 3 6 1 2 9 3 1 9 4 7 8 4 5 0
Y : 3 6 1 0 6 3 2 0 6 1 5 5 4 7 6 5 6 9 3 7 4 5 2 5 4 7 4 4 3 0 7 8 6 8 8 4 3 1 4 9 2 0 6 8 9 2 6 6 4 9
Z : 5 0 4 8 7 1 7 2 7 2 2 6 1 0 6 1 5 9 4 9 0 9 1 7 7 1 1 5 9 7 7 6 7 3 6 5 6 3 9 4 8 1 2 9 3 9 0 8 8 5
Longest Common Subsequence W :
6 1 0 6 0 1 7 9 7 7 3 6 1 2 9 9
6 1 0 6 0 1 7 9 7 7 3 6 3 1 9 8

Length of W : 16
```

(值限制在 0~9)

觀察兩個結果發現值限制在 0~9 時，相比於值限制在 0~49，它會有更長的 Longest Common Subsequence，除此之外，也會發現值限制在 0~9 時，有更多組的 LCS。

C. 探討複雜度

1. **窮舉法的時間複雜度:**最直接簡單的暴力窮舉法是將三個 sequence 的所有 subsequence 全部列出來，在找尋最大的共同 subsequence，假設 3 個 sequence 的長度分別為 n1,n2,n3，而他們的 subsequence 分別為2^{n1}, 2^{n2}, 2^{n3}，因此我們可以得知他的時間複雜度為 O($2^{n1} * 2^{n2} * 2^{n3}$)，用此方法複雜度非常高，特別是當 n1,n2,n3 都非常大時，可能不太適用。
2. **動態規劃的時間複雜度:**老師上課講的方法是利用動態規劃去解決此問題，先將較小問題的答案儲存，在透過小問題的答案去解決更大問題的答案，如此可以大幅減少時間複雜度，為 O(n1 * n2 *n3)。

```
50      for(int i=1;i<=m;i++){
51          for(int j=1;j<=n;j++){
52              for(int k=1;k<=q;k++){
53                  if(x[i]==y[j]&&y[j]==z[k]){
54                      C[i][j][k]=C[i-1][j-1][k-1]+1;
55                      b[i][j][k]='A';
56                  }
57                  else if(C[i-1][j][k]>=C[i][j-1][k] && C[i-1][j][k]>=C[i][j][k-1]){
58                      C[i][j][k]=C[i-1][j][k];
59                      b[i][j][k]='B';
60                  }
61                  else if(C[i][j-1][k]>=C[i-1][j][k] && C[i][j-1][k]>=C[i][j][k-1]){
62                      C[i][j][k]=C[i][j-1][k];
63                      b[i][j][k]='C';
64                  }
65                  else{
66                      C[i][j][k]=C[i][j][k-1];
67                      b[i][j][k]='D';
68                  }
69              }
70          }
71      }
```

(遞迴關係式)