**10202449 Po-Ren Pan**

**Longest Common Sequence Reference Book Solution**

* **Psuedo Code**

Input: string A, string B

Output: Array L

m <- A.size();

n <- B.size();

L[m+1][n+1];

for(int i=0;i<=m;i++)

dp[i][0] = 0;

for(int j=0;j<=n;j++)

dp[0][j] = 0;

int max\_n = 1;

for(int i=1;i<=m;i++)

{

for(int j=1;j<=n;j++)

{

if(A[i] == A[j])

dp[i][j] <- L[i-1][j-1] + 1;

else

dp[i][j] <- max(L[i-1][j], L[i][j-1]);

}

}

return L

* **Complexity**

Time complexity : O(mn) This is because dynamic process fills in the whole m\*n table.

Space complexity: O(mn) dp array m\*n

**Longest Common Sequence Linear Memory Solution**

* **Psuedo Code**

Input: string A, string B

Output: Array L

m <- A.size();

n <- B.size();

L [2][n+1];

for(int j=0;j<=n;j++)

L[1][j] <- 0

for(int i=0;i<=m;i++){

for(int j=0;j<=n;j++){

// copy to first row

for(int j=0;j<=n;j++)

dp[0][j] = dp[1][j];

for(int k=1;k<=n;k++) {

if(A[i-1] == B[k-1])

dp[1][k] = dp[0][k-1] + 1;

else

dp[1][k] = max(dp[0][k], dp[1][k-1]);

}

}

}

* **Complexity**

Time complexity : O(mn) This is because dynamic process goes through all characters in two string A, B

Space complexity: O(n) Only need 2 rows to finish LCS