DAA Lab

Practical 5

Name: Madhura Mahatme

Section: A1-B3

Roll no.: 36

Git Link: https://github.com/mahatmemadhura4-bot/DAA-Lab-Pract5

Aim:

Implement Longest Common Subsequence (LCS) algorithm to find the length and LCS for DNA sequences.

Problem Statement:

DNA sequences can be viewed as strings of A, C, G, and T characters, which represent nucleotides. Finding the similarities between two DNA sequences are an important computation performed in bioinformatics.

[Note that a subsequence might not include consecutive elements of the original sequence.]

TASK-1:

Find the similarity between the given X and Y sequence.

X=AGCCCTAAGGGCTACCTAGCTT

Y= GACAGCCTACAAGCGTTAGCTTG

Code:

```
#include <stdio.h>

#include <string.h>

#define MAX 100
```

```
void lcs(char a[], char b[], int c[MAX][MAX], char dir[MAX][MAX], int m, int n) {
    for(int i = 0; i <= m; i++){
        c[i][0] = 0;
    }
    for(int j = 0; j <= n; j++){
        c[0][j] = 0;
    }

    for(int i = 1; i <= m; i++){
        for(int j = 1; j <= n; j++){
            if (a[i-1] == b[j-1]) {
                c[i][j] = c[i-1][j-1] + 1;
                dir[i][j] = 'D';
            }
            else if(c[i-1][j] >= c[i][j-1]){
                c[i][j] = c[i-1][j];
                dir[i][j] = 'U';
            }
            else{
                c[i][j] = c[i][j-1];
                dir[i][j] = 'L';
            }
        }
    }
}
void print_matrix(int c[MAX][MAX], char dir[MAX][MAX], int m, int n, char a[], char b[])
{
    printf("\nCost Matrix with Directions:\n\n   ");
    for(int j = 0; j < n; j++){
```

```
        printf("   %c", b[j]);
    }
    printf("\n");

    for (int i = 0; i <= m; i++){
        if (i==0) printf("  ");
        else printf("%c ", a[i-1]);

        for (int j = 0; j <= n; j++){
            if (i==0 || j==0)
                printf("  %2d ", c[i][j]);
            else
                printf(" %2d%c ", c[i][j], dir[i][j]);
        }
        printf("\n");
    }
}
void print_lcs(char a[], char b[], char dir[MAX][MAX], int i, int j){
    if(i==0 || j==0)
        return;
    if(dir[i][j] == 'D'){
        print_lcs(a, b, dir, i-1, j-1);
        printf("%c", a[i-1]);
    } else if (dir[i][j] == 'U'){
        print_lcs(a, b, dir, i-1, j);
    } else {
        print_lcs(a, b, dir, i, j-1);
    }
}
```

```
int main(){
    char X[MAX], Y[MAX];
    int c[MAX][MAX];
    char dir[MAX][MAX];

    printf("Enter first sequence (X): ");
    scanf("%s", X);

    printf("Enter second sequence (Y): ");
    scanf("%s", Y);

    int m = strlen(X);
    int n = strlen(Y);

    lcs(X, Y, c, dir, m, n);

    print_matrix(c, dir, m, n, X, Y);

    printf("\nFinal cost (Length of LCS): %d\n", c[m][n]);

    printf("LCS: ");
    print_lcs(X, Y, dir, m, n);
    printf("\n");

    return 0;
}
```

```c
#include <stdio.h>
#include <string.h>
#define MAX 100
void lcs(char a[], char b[], int c[MAX][MAX], char dir[MAX][MAX], int m, int n) {
    for(int i = 0; i <= m; i++){
        c[i][0] = 0;
    }
    for(int j = 0; j <= n; j++){
        c[0][j] = 0;
    }
    for(int i = 1; i <= m; i++){
        for(int j = 1; j <= n; j++){
            if(a[i-1] == b[j-1]){
                c[i][j] = c[i-1][j-1] + 1;
                dir[i][j] = 'D';
            }
            else if(c[i-1][j] >= c[i][j-1]){
                c[i][j] = c[i-1][j];
                dir[i][j] = 'U';
            }
            else{
                c[i][j] = c[i][j-1];
                dir[i][j] = 'L';
            }
        }
    }
}
void print_matrix(int c[MAX][MAX], char dir[MAX][MAX], int m, int n, char a[], char b[]){
    printf("\nCost Matrix with Directions:\n\n   ");
    for(int j = 0; j < n; j++){
        printf("   %c", b[j]);
    }
    printf("\n");

    for (int i = 0; i <= m; i++){
        if (i==0) printf("  ");
        else printf("%c ", a[i-1]);
```

```c

        for (int j = 0; j <= n; j++){
            if (i==0 || j==0)
                printf("  %2d ", c[i][j]);
            else
                printf(" %2d%c ", c[i][j], dir[i][j]);
        }
        printf("\n");
    }
}
void print_lcs(char a[], char b[], char dir[MAX][MAX], int i, int j){
    if(i==0 || j==0)
        return;
    if(dir[i][j] == 'D'){
        print_lcs(a, b, dir, i-1, j-1);
        printf("%c", a[i-1]);
    }
    else if(dir[i][j] == 'U'){
        print_lcs(a, b, dir, i-1, j);
    }
    else{
        print_lcs(a, b, dir, i, j-1);
    }
}

int main(){
    char X[MAX], Y[MAX];
    int c[MAX][MAX];
    char dir[MAX][MAX];

    printf("Enter first sequence (X): ");
    scanf("%s", X);

```

```c
    printf("Enter second sequence (Y): ");
    scanf("%s", Y);

    int m = strlen(X);
    int n = strlen(Y);

    lcs(X, Y, c, dir, m, n);

    print_matrix(c, dir, m, n, X, Y);

    printf("\nFinal cost (Length of LCS): %d\n", c[m][n]);

    printf("LCS: ");
    print_lcs(X, Y, dir, m, n);
    printf("\n");

    return 0;
}


```

Output:

```
Enter first sequence (X): AGCCCTAAGGGCTACCTAGCTT
Enter second sequence (Y): GACAGCCTACAAGCGTTAGCTTG

Cost Matrix with Directions:

      G   A   C   A   G   C   C   T   A   C   A   A   G   C   G   T   T   A   G   C   T   T   G
      0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
A     0   0U  1D  1L  1D  1L  1L  1L  1L  1D  1L  1D  1D  1L  1L  1L  1L  1L  1D  1L  1L  1L  1L  1L
G     0   1D  1U  1U  1U  2D  2L  2L  2L  2L  2L  2L  2L  2D  2L  2D  2L  2L  2L  2D  2L  2L  2L  2D
C     0   1U  1U  2D  2L  2U  3D  3D  3L  3L  3D  3L  3L  3L  3D  3L  3L  3L  3L  3L  3D  3L  3L  3L
C     0   1U  1U  2D  2U  2U  3D  4D  4L  4L  4D  4L  4L  4L  4D  4L  4L  4L  4L  4L  4D  4L  4L  4L
C     0   1U  1U  2D  2U  2U  3D  4D  4U  4U  5D  5L  5L  5L  5D  5L  5L  5L  5L  5L  5D  5L  5L  5L
T     0   1U  1U  2U  2U  2U  3U  4U  5D  5L  5U  5U  5U  5U  5U  5U  6D  6D  6L  6L  6L  6D  6D  6L
A     0   1U  2D  2U  3D  3L  3U  4U  5U  6D  6L  6D  6D  6L  6L  6L  6U  6U  7D  7L  7L  7L  7L  7L
A     0   1U  2D  2U  3D  3U  3U  4U  5U  6D  6U  7D  7D  7L  7L  7L  7L  7L  7D  7U  7U  7U  7U  7U
G     0   1D  2U  2U  3U  4D  4L  4U  5U  6U  6U  7U  7U  8D  8L  8D  8L  8L  8L  8D  8L  8L  8L  8D
G     0   1D  2U  2U  3U  4D  4U  4U  5U  6U  6U  7U  7U  8D  8U  9D  9L  9L  9L  9D  9L  9L  9L  9D
G     0   1D  2U  2U  3U  4D  4U  4U  5U  6U  6U  7U  7U  8D  8U  9D  9U  9U  9U  10D 10L 10L 10L 10D
C     0   1U  2U  3D  3U  4U  5D  5D  5U  6U  7D  7U  7U  8U  9D  9U  9U  9U  9U  10U 11D 11L 11L 11L
T     0   1U  2U  3U  3U  4U  5U  5U  6D  6U  7U  7U  7U  8U  9U  9U  10D 10D 10L 10U 11U 12D 12D 12L
A     0   1U  2D  3U  4D  4U  5U  5U  6U  7D  7U  8D  8D  8U  9U  9U  10U 10U 11D 11L 11U 12U 12U 12U
C     0   1U  2U  3D  4U  4U  5D  6D  6U  7U  8D  8U  8U  8U  9D  9U  10U 10U 11U 11U 12D 12U 12U 12U
C     0   1U  2U  3D  4U  4U  5D  6D  6U  7U  8D  8U  8U  8U  9D  9U  10U 10U 11U 11U 12D 12U 12U 12U
T     0   1U  2U  3U  4U  4U  5U  6U  7D  7U  8U  8U  8U  8U  9U  9U  10D 11D 11U 11U 12U 13D 13D 13L
A     0   1U  2D  3U  4D  4U  5U  6U  7U  8D  8U  9D  9D  9L  9U  9U  10U 11U 12D 12L 12U 13U 13U 13U
G     0   1D  2U  3U  4U  5D  5U  6U  7U  8U  8U  9U  9U  10D 10L 10D 10U 11U 12U 13D 13L 13U 13U 14D
C     0   1U  2U  3D  4U  5U  6D  6D  7U  8U  9D  9U  9U  10U 11D 11L 11L 11U 12U 13U 14D 14L 14L 14U
T     0   1U  2U  3U  4U  5U  6U  6U  7D  8U  9U  9U  9U  10U 11U 11U 12D 12D 12U 13U 14U 15D 15D 15L
T     0   1U  2U  3U  4U  5U  6U  6U  7D  8U  9U  9U  9U  10U 11U 11U 12D 13D 13L 13U 14U 15D 16D 16L

Final cost (Length of LCS): 16
LCS: AGCCCAAGGTTAGCTT
PS C:\Users\Madhura\OneDrive\Desktop\C>
```

TASK-2:

Find the longest repeating subsequence (LRS). Consider it as a variation of the

longest common subsequence (LCS) problem.

Let the given string be S. You need to find the LRS within S. To use the LCS framework, you effectively compare S with itself. So, consider string1 = S and string2 = S.

Example:

AABCBDC

LRS= ABC or ABD

Code:

```
#include <stdio.h>
#include <string.h>
#define MAX 100
void lrs(char s[], int c[MAX][MAX], int m){
    for (int i=0; i <= m; i++) {
        c[i][0] = 0;
        c[0][i] = 0;
    }
    for (int i = 1; i <= m; i++){
        for (int j = 1; j <= m; j++){
            if (s[i-1]==s[j-1] && i != j){
                c[i][j]=c[i-1][j-1] + 1;
            }
            else{
                c[i][j] = (c[i-1][j] > c[i][j-1]) ? c[i-1][j] : c[i][j-1];
            }
        }
    }
}
void print_lrs(char s[], int c[MAX][MAX], int i, int j){
    if (i==0 || j==0) return;

    if (s[i-1] == s[j-1] && i != j){
        print_lrs(s, c, i-1, j-1);
        printf("%c", s[i-1]);
    }
    else if(c[i-1][j] > c[i][j-1]){
        print_lrs(s, c, i-1, j);
```

```
    }
    else{
        print_lrs(s, c, i, j-1);
    }
}

int main(){
    char S[MAX];
    int c[MAX][MAX];
    printf("Enter the string: ");
    scanf("%s", S);

    int m = strlen(S);
    lrs(S, c, m);

    printf("\nLength of Longest Repeating Subsequence: %d\n", c[m][m]);
    printf("LRS: ");
    print_lrs(S, c, m, m);
    printf("\n");

    return 0;
}
```

```c
#include <stdio.h>
#include <string.h>
#define MAX 100
void lrs(char s[], int c[MAX][MAX], int m){
    for (int i=0; i <= m; i++) {
        c[i][0] = 0;
        c[0][i] = 0;
    }
    for (int i = 1; i <= m; i++){
        for (int j = 1; j <= m; j++){
            if (s[i-1]==s[j-1] && i != j){
                c[i][j]=c[i-1][j-1] + 1;
            }
            else{
                c[i][j] = (c[i-1][j] > c[i][j-1]) ? c[i-1][j] : c[i][j-1];
            }
        }
    }
}
void print_lrs(char s[], int c[MAX][MAX], int i, int j){
    if (i==0 || j==0) return;

    if (s[i-1] == s[j-1] && i != j){
        print_lrs(s, c, i-1, j-1);
        printf("%c", s[i-1]);
    }
    else if(c[i-1][j] > c[i][j-1]){
        print_lrs(s, c, i-1, j);
    }
    else{
        print_lrs(s, c, i, j-1);
    }
}

int main(){
    char S[MAX];
    int c[MAX][MAX];
    printf("Enter the string: ");
    scanf("%s", S);

    int m = strlen(S);
    lrs(S, c, m);

    printf("\nLength of Longest Repeating Subsequence: %d\n", c[m][m]);
    printf("LRS: ");
    print_lrs(S, c, m, m);
    printf("\n");

    return 0;
}
```

Output:

```
Enter the string: AABCBDC

Length of Longest Repeating Subsequence: 3
LRS: ABC
PS C:\Users\Madhura\OneDrive\Desktop\C>
```

LeetCode Assesment: