**Experiment-7**

**Title: Dynamic Programming-II**

1. **Implement Floyd’s algorithm for the All-Pairs-Shortest-Paths problem.**

Code:

#include<stdio.h>

#define V 4

#define INF 99999

void printSolution(int dist[][V]);

void floydWarshall (int graph[][V])

{

int dist[V][V], i, j, k;

for (i = 0; i < V; i++)

for (j = 0; j < V; j++)

dist[i][j] = graph[i][j];

for (k = 0; k < V; k++)

{

for (i = 0; i < V; i++)

{

for (j = 0; j < V; j++)

{

if (dist[i][k] + dist[k][j] < dist[i][j])

dist[i][j] = dist[i][k] + dist[k][j];

}

}

}

printSolution(dist);

}

void printSolution(int dist[][V])

{

printf ("The following matrix shows the shortest distances"

" between every pair of vertices \n");

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

{

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

{

if (dist[i][j] == INF)

printf("%7s", "INF");

else

printf ("%7d", dist[i][j]);

}

printf("\n");

}

}

int main()

{

int graph[V][V] = { {0, 5, INF, 10},

{INF, 0, 3, INF},

{INF, INF, 0, 1},

{INF, INF, INF, 0}

};

floydWarshall(graph);

return 0;

}

1. **Given two sequences, find the length of longest subsequence present in both of them. A subsequence is a sequence that appears in the same relative order, but not necessarily contiguous. For example, “abc”, “abg”, “bdf”, “aeg”, ‘”acefg” etc are subsequences of “abcdefg”.**

Code:

#include<stdio.h>

#include<string.h>

int max(int a, int b);

int lcs( char \*X, char \*Y, int m, int n )

{

if (m == 0 || n == 0)

return 0;

if (X[m-1] == Y[n-1])

return 1 + lcs(X, Y, m-1, n-1);

else

return max(lcs(X, Y, m, n-1), lcs(X, Y, m-1, n));

}

int max(int a, int b)

{

return (a > b)? a : b;

}

int main()

{

char X[100],Y[100];

scanf("%s",&X);

scanf("%s",&Y);

int m = strlen(X);

int n = strlen(Y);

printf("Length of LCS is %d", lcs( X, Y, m, n ) );

return 0;

}