Experiment 14

Write a program to implement the Dynamic Floyd Warshwall Algorithm to solve all pair shortest path problem.

Program:-

```
#include <stdio.h>
#include <conio.h>
#include <time.h>
int min(int a, int b);
void floyds(int p[10][10], int n) {
   int i, j, k;
   for (k = 1; k <= n; k++) {
      for (i = 1; i <= n; i++) {
         for (j = 1; j <= n; j++) {
            if (i == j) {
               p[i][j] = 0;
            } else {
               p[i][j] = min(p[i][j], p[i][k] + p[k][j]);
            }
         }
      }
   }
}
int min(int a, int b) {
   if (a < b) {
      return (a);
   } else {
      return (b);
   }
}
void main() {
   double time;
```

```
clock_t start, end;
int p[100][100],w, n, e, u, v,i,j;
printf("\n Enter the number of vertices:");
scanf("%d", &n);
printf("\n Enter the number of edges: \n");
scanf("%d", &e);
for (i = 1; i <= n; i++) {
   for (j = 1; j <= n; j++) {
      p[i][j] = 999;
   }
}
for (i = 1; i <= e; i++) {
   printf("\n Enter the end vertices of edge %d with its weight \n", i);
   scanf("%d%d%d", &u, &v, &w);
   p[u][v] = w;
}
printf("\n Note: 999 = infinity\n");
printf("\n Matrix of input data:\n");
for (i = 1; i <= n; i++) {
   for (j = 1; j <= n; j++) {
      printf("%d \t", p[i][j]);
   }
   printf("\n");
}
start = clock();
floyds(p, n);
printf("\n Transitive closure:\n");
for (i = 1; i <= n; i++) {
   for (j = 1; j <= n; j++) {
      printf("%d \t", p[i][j]);
```

```
        }
        printf("\n");
    }
    printf("\n The shortest paths are:\n");
    for (i = 1; i <= n; i++) {
        for (j = 1; j <= n; j++) {
            if (i != j) {
                printf("\n <%d,%d> = %d", i, j, p[i][j]);
            }
        }
    }
    end = clock();
    time = ((double) (end - start) * 1000) / CLOCKS_PER_SEC;
    printf("\n Time = %lf milliseconds", time);
}
```

Result Analysis and Discussion:

```
 Enter the number of vertices:4

 Enter the number of edges:
7

 Enter the end vertices of edge 1 with its weight
2 3 1

 Enter the end vertices of edge 2 with its weight
1 4 2

 Enter the end vertices of edge 3 with its weight
3 7 5

 Enter the end vertices of edge 4 with its weight
8 6 2
```

```
 Enter the end vertices of edge 5 with its weight
5 4 6

 Enter the end vertices of edge 6 with its weight
4 4 4

 Enter the end vertices of edge 7 with its weight
9 9 9

 Note: 999 = infinity

 Matrix of input data:
999     999     999     2
999     999     1       999
999     999     999     999
999     999     999     4

 Transitive closure:
999     999     999     2
999     999     1       999
999     999     999     999
999     999     999     4

The shortest paths are:

<1,2> = 999
<1,3> = 999
<1,4> = 2
<2,1> = 999
<2,3> = 1
<2,4> = 999
<3,1> = 999
<3,2> = 999
<3,4> = 999
<4,1> = 999
<4,2> = 999
<4,3> = 999
Time = 4.000000 milliseconds
```

This experiment has been conducted in a 64-bit system with 16 GB RAM and Processor 12th Gen Intel(R) Core (TM) i5-12500H 3.10 GHz. The algorithm is implemented in C programming language in Visual Studio Code 1.83.1 Code Editor. The time taken by this algorithm for 4 vertices and 7 edges is 4 milliseconds.

Conclusion:

The running time of Dynamic Floyd Warshwall Algorithm to solve all pair shortest path problem is analyzed as O(n3).