**Experiment 3.3**

**Student Name:   Gaurav Kumar                                UID: 22MCC20177**

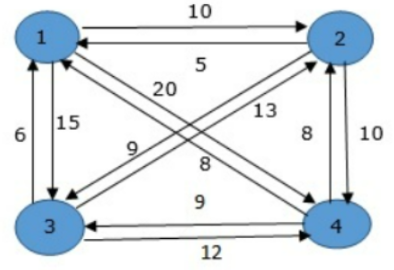
**Branch:   CC-DevOps                                                     Section/Group:- 1/B**

**Semester:   One                                                               Date of Performance: 03/01/2023**

**Subject Name:- Design & Analysis of Algorithms Lab                   Subject Code: 22CAP-646**

1. **Task to be done:**

**Implement any scheme to find the optimal solution for the Traveling Salesperson problem and then solve the same problem instance using any approximation algorithm and determine the error in the approximation.**



1. **Steps for experiment/practical: copy and paste your code here/screenshots.**

#include <stdio.h>

int a[10][10], n, visit[10];

int cost\_opt = 0, cost\_apr = 0;

int least\_apr(int c);

int least\_opt(int c);

void mincost\_opt(int city)

{

    int i, ncity;

    visit[city] = 1;

    printf("%d-->", city);

    ncity = least\_opt(city);

    if (ncity == 999)

    {

        ncity = 1;

        printf("%d", ncity);

        cost\_opt += a[city][ncity];

        return;

    }

    mincost\_opt(ncity);

}

void mincost\_apr(int city)

{

    int i, ncity;

    visit[city] = 1;

    printf("%d-->", city);

    ncity = least\_apr(city);

    if (ncity == 999)

    {

        ncity = 1;

        printf("%d", ncity);

        cost\_apr += a[city][ncity];

        return;

    }

    mincost\_apr(ncity);

}

int least\_opt(int c)

{

    int i, nc = 999;

    int min = 999, kmin = 999;

    for (i = 1; i <= n; i++)

    {

        if ((a[c][i] != 0) && (visit[i] == 0))

            if (a[c][i] < min)

            {

                min = a[i][1] + a[c][i];

                kmin = a[c][i];

                nc = i;

            }

    }

    if (min != 999)

        cost\_opt += kmin;

    return nc;

}

int least\_apr(int c)

{

    int i, nc = 999;

    int min = 999, kmin = 999;

    for (i = 1; i <= n; i++)

    {

        if ((a[c][i] != 0) && (visit[i] == 0))

            if (a[c][i] < kmin)

            {

                min = a[i][1] + a[c][i];

                kmin = a[c][i];

                nc = i;

            }

    }

    if (min != 999)

        cost\_apr += kmin;

    return nc;

}

int main()

{

    int i, j;

    printf("Enter No. of cities:\n");

    scanf("%d", &n);

    printf("Enter the cost matrix\n");

    for (i = 1; i <= n; i++)

    {

        printf("Enter elements of row:%d\n", i);

        for (j = 1; j <= n; j++)

            scanf("%d", &a[i][j]);

        visit[i] = 0;

    }

    printf("The cost list is \n");

    for (i = 1; i <= n; i++)

    {

        printf("\n\n");

        for (j = 1; j <= n; j++)

            printf("\t%d", a[i][j]);

    }

    printf("\n\n Optimal Solution :\n");

    printf("\n The path is :\n");

    mincost\_opt(1);

    printf("\n Minimum cost:");

    printf("%d", cost\_opt);

    printf("\n\n Approximated Solution :\n");

    for (i = 1; i <= n; i++)

        visit[i] = 0;

    printf("\n  The path is :\n");

    mincost\_apr(1);

    printf("\nMinimum cost:");

    printf("%d", cost\_apr);

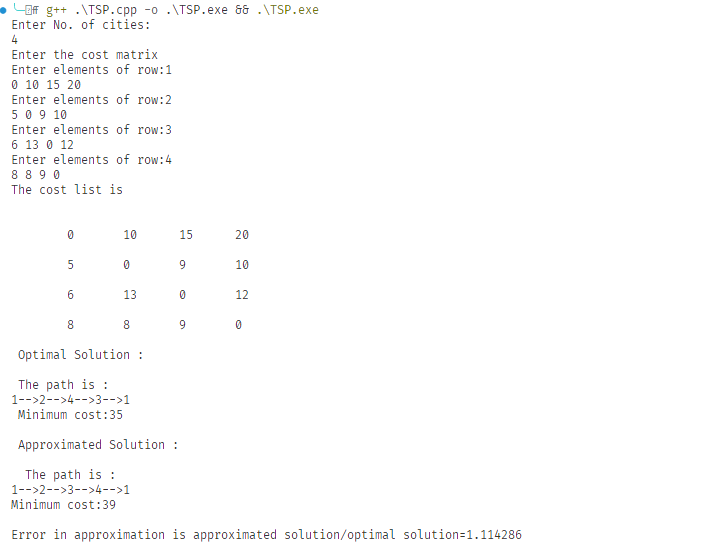
    printf("\n\nError in approximation is approximated solution/optimal solution=%f",

           (float)cost\_apr / cost\_opt);

    return 0;

}

1. **Output (screenshots)**

****

**Evaluation Grid:**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. | Parameters | Marks Obtained | Maximum Marks |
| 1. | Demonstration and Performance  (Quiz) |  | 22 |
| 2. | Worksheet |  | 8 |