**FLOYD WARSHAL**

#include <stdio.h>

#include <limits.h>

#include <stdlib.h>

#define min(a, b) ((a) < (b) ? (a) : (b))

void printMatrix(int \*\*graph, int n)

{

  int i, j;

  for (i = 0; i < n; i++)

  {

    for (j = 0; j < n; j++)

    {

      if (graph[i][j] == INT\_MAX)

      {

        printf("I ");

      }

      else

      {

        printf("%d ", graph[i][j]);

      }

    }

    printf("\n");

  }

}

void floydWarshal(int \*\*graph, int n)

{

  int i, j, k;

  for (k = 0; k < n; k++)

  {

    for (i = 0; i < n; i++)

    {

      for (j = 0; j < n; j++)

      {

        if (graph[i][k] != INT\_MAX && graph[k][j] != INT\_MAX)

        {

          graph[i][j] = min(graph[i][j], graph[i][k] + graph[k][j]);

        }

      }

    }

    printf("\nMatrix after %d iteration:\n", k + 1);

    printMatrix(graph, n);

  }

}

int main()

{

  int n, e, i, v1, v2, w, j;

  int \*\*graph = (int \*\*)malloc(n \* sizeof(int \*));

  FILE \*file = fopen("floyd.txt", "r");

  if (file == NULL)

  {

    printf("Error opening file");

    return 0;

  }

  fscanf(file, "%d %d", &n, &e);

  for (i = 0; i < n; i++)

  {

    graph[i] = (int \*)malloc(n \* sizeof(int));

  }

  for (i = 0; i < n; i++)

  {

    for (j = 0; j < n; j++)

    {

      if (i == j)

      {

        graph[i][j] = 0;

      }

      else

      {

        graph[i][j] = INT\_MAX;

      }

    }

  }

  for (i = 0; i < e; i++)

  {

    fscanf(file, "%d %d %d", &v1, &v2, &w);

    graph[v1 - 1][v2 - 1] = w;

  }

  fclose(file);

  printf("\nInitial matrix:\n");

  printMatrix(graph, n);

  floydWarshal(graph, n);

  return 0;

}

FLOYD.TXT

\*\*FIRST LINE VERTEX AND EDGE\*\*

\*\*GRAPH STARTS WITH 1\*\*

4 7

1 2 3

1 4 2

2 1 5

2 3 6

3 4 4

4 2 -1

4 3 -3