WEEK 7 ADA LAB

Q) Implement floyds algorithm using a c program

SOURCE CODE:-

// Floyd-Warshall Algorithm in C

#include <stdio.h>

// defining the number of vertices

#define nV 4

#define INF 999

void printMatrix(int matrix[][nV]);

// Implementing floyd warshall algorithm

void floydWarshall(int graph[][nV]) {

int matrix[nV][nV], i, j, k;

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

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

matrix[i][j] = graph[i][j];

// Adding vertices individually

for (k = 0; k < nV; k++) {

for (i = 0; i < nV; i++) {

for (j = 0; j < nV; j++) {

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

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

}

}

}

printMatrix(matrix);

}

void printMatrix(int matrix[][nV]) {

for (int i = 0; i < nV; i++) {

for (int j = 0; j < nV; j++) {

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

printf("%4s", "INF");

else

printf("%4d", matrix[i][j]);

}

printf("\n");

}

}

int main() {

int graph[nV][nV] = {{0, INF, 3, INF},

{2, 0, INF, INF},

{INF, 7, 0, 1},

{6, INF, INF, 0}};

floydWarshall(graph);

}

OUTPUT:-



Q) Implement knapsack problem using c program

SOURCE CODE:-

#include <stdio.h>

int max(int a, int b) { return (a > b) ? a : b; }

int knapSack(int W, int wt[], int val[], int n)

{

if (n == 0 || W == 0)

return 0;

if (wt[n - 1] > W)

return knapSack(W, wt, val, n - 1);

else

return max(

val[n - 1]

+ knapSack(W - wt[n - 1], wt, val, n - 1),

knapSack(W, wt, val, n - 1));

}

int main()

{

int profit[10];

int weight[10];

int W,k;

printf("Enter the knapsack capacity:\n");

scanf("%d",&W);

printf("Enter the number of weights:\n");

scanf("%d",&k);

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

{

printf("Enter the weight and profit:\n");

scanf("%d %d",&weight[i],&profit[i]);

}

printf("The total profit is :\n");

printf("%d", knapSack(W, weight, profit, k));

return 0;

}

OUTPUT:-

