**Assignment - 6**

**Multimedia Lab**

**Name – Shishu**  **Registration No- 2020CA089**

Ques - Implementation of JPEG image compression using Discrete Cosine Transform.

## **Code:**

// CPP program to perform discrete cosine transform

#include <bits/stdc++.h>

using namespace std;

#define pi 3.142857

const int m = 8, n = 8;

// Function to find discrete cosine transform and print it

int dctTransform(int matrix[][n])

{

int i, j, k, l;

// dct will store the discrete cosine transform

float dct[m][n];

float ci, cj, dct1, sum;

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

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

// ci and cj depends on frequency as well as

// number of row and columns of specified matrix

if (i == 0)

ci = 1 / sqrt(m);

else

ci = sqrt(2) / sqrt(m);

if (j == 0)

cj = 1 / sqrt(n);

else

cj = sqrt(2) / sqrt(n);

// sum will temporarily store the sum of

// cosine signals

sum = 0;

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

for (l = 0; l < n; l++) {

dct1 = matrix[k][l] \*

cos((2 \* k + 1) \* i \* pi / (2 \* m)) \*

cos((2 \* l + 1) \* j \* pi / (2 \* n));

sum = sum + dct1;

}

}

dct[i][j] = ci \* cj \* sum;

}

}

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

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

printf("%f\t", dct[i][j]);

}

printf("\n");

}

}

// Driver code

int main()

{

int matrix[m][n] = { { 255, 255, 255, 255, 255, 255, 255, 255 },

{ 255, 255, 255, 255, 255, 255, 255, 255 },

{ 255, 255, 255, 255, 255, 255, 255, 255 },

{ 255, 255, 255, 255, 255, 255, 255, 255 },

{ 255, 255, 255, 255, 255, 255, 255, 255 },

{ 255, 255, 255, 255, 255, 255, 255, 255 },

{ 255, 255, 255, 255, 255, 255, 255, 255 },

{ 255, 255, 255, 255, 255, 255, 255, 255 } };

dctTransform(matrix);

return 0;

}

//This code is contributed by SoumikMondal

## **Output:**

