**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное автономное образовательное учреждение

высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт кибернетики

Направление (специальность) Информационные системы и технологии

Кафедра информационных систем и технологий

**Отчет по лабораторной работе №3 "Алгоритм JPEG"**по дисциплине Основы теории передачи информации

Вариант 14

Выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Смирнов П.О.

Студент группы 8И5Б

Проверил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.К. Стоянов

Доцент кафедры ИСиТ

Томск 2017

**Цель работы:**

Составить на языке программу, выполняющую следующие функции:

* формирование ДКП матрицы по формулам (5). Вычисление осуществлять с точностью 6 знаков после запятой (точки);
* формирование транспонированной ДКП матрицы;
* формирование из исходной матрицы изображения (дает преподаватель) PDCT-матрицу. Напомним, что перед выполнением ДКП нужно из всех элементов матрицы изображения вычесть число 128; в этой части программы, предусмотреть возможность использования или матрицы ДКП, получаемой по формуле (5), или матрицы ДКП получаемой по формуле (6);
* составление матрицы квантования для качества, задаваемого с клавиатуры;
* реализация квантования, т. е. получениеPQDCT;
* преобразование PQDCT в матрицу распакованного изображения.

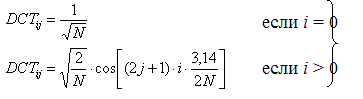


Рисунок 1 – Преобразование ДКП

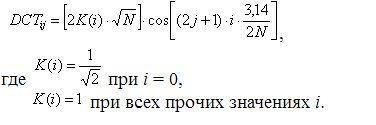


Рисунок 2 – Формула 6

Вариант задания:

138 135 136 130 128 133 142 150

136 135 138 136 139 145 152 161

140 138 139 146 152 158 161 168

140 138 141 146 154 163 169 170

140 138 142 147 153 162 166 166

137 135 138 143 149 158 163 163

136 132 130 135 142 150 155 158

138 132 126 126 133 136 138 143

*Результаты работы программы:*

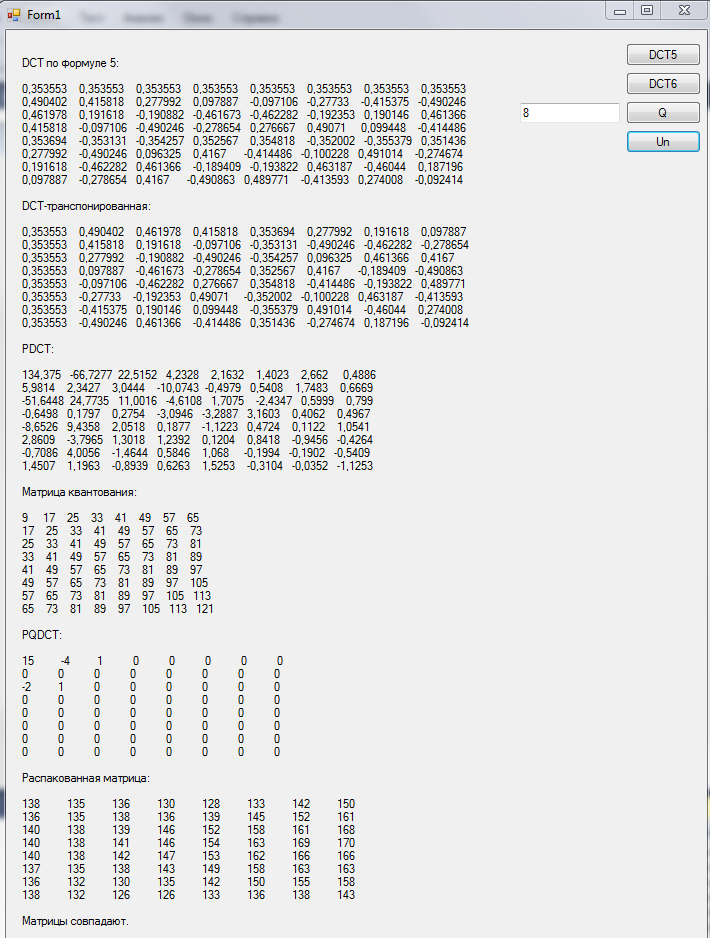


Рисунок 3 – Результаты по формуле 5

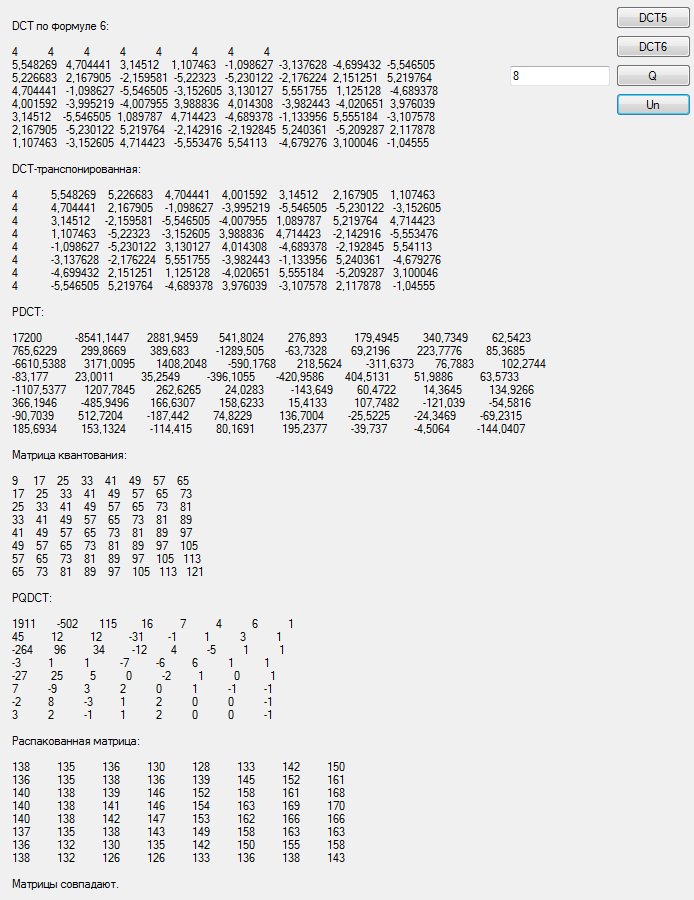


Рисунок 4 – Результаты по формуле 6

**Вывод:**

В ходе лабораторной работы был освоен алгоритм сжатия изображений по стандарту JPEG. Было программно реализовано дискретное косинусное преобразование.

***Листинг:***

namespace JPEGform

{

public partial class Form1 : Form

{

static string[] FromFile = File.ReadAllLines(@"C:\Users\User\Desktop\File.txt", Encoding.Default);

static int size = FromFile.Length;

static double[,] P = new double[size, size];

static double[,] DCT;

static double[,] DCTT;

static double[,] PDCT;

static double[,] QUANT;

static double[,] PQDCT;

public Form1()

{

InitializeComponent();

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

{

string[] row = FromFile[i].Split(' ');

for (int j = 0; j < size; j++)

P[i, j] = Convert.ToDouble(row[j]) - 128.0;

}

}

private void button1\_Click(object sender, EventArgs e)

{

label1.Text = "";

PDCT = new double[size, size];

DCT = new double[size, size];

DCTT = new double[size, size];

label1.Text += "DCT по формуле 5:\n";

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

{

for (int j = 0; j < size; j++)

{

if (i == 0)

DCT[i, j] = 1.0 / Math.Sqrt(size);

else

DCT[i, j] = Math.Sqrt(2.0 / size) \* Math.Cos((2.0 \* j + 1) \* i \* (3.14 / (2.0 \* size)));

label1.Text += String.Format("{0,-11} ", Math.Round(DCT[i, j],6));

}

label1.Text += "\n";

}

label1.Text += "DCT-транспонированная:\n";

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

{

for (int j = 0; j < size; j++)

{

DCTT[i, j] = DCT[j, i];

label1.Text += String.Format("{0,-11} ", Math.Round(DCTT[i, j],6));

}

label1.Text += "\n";

}

double[,] PDCT1 = new double[size, size];

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

{

for (int j = 0; j < size; j++)

{

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

{

PDCT1[i, j] += DCT[i, k] \* P[k, j];

}

}

}

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

{

for (int j = 0; j < size; j++)

{

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

{

PDCT[i, j] += PDCT1[i, k] \* DCTT[k, j];

}

}

}

label1.Text += "PDCT:\n";

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

{

for (int j = 0; j < size; j++)

label1.Text += String.Format("{0,-10}", Math.Round(PDCT[i, j],4));

label1.Text += "\n";

}

}

private void button2\_Click(object sender, EventArgs e)

{

label1.Text = "";

PDCT = new double[size, size];

DCT = new double[size, size];

DCTT = new double[size, size];

label1.Text += "DCT по формуле 6:\n";

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

{

for (int j = 0; j < size; j++)

{

if (i == 0)

DCT[i, j] = Math.Sqrt(2.0 \* size) \* Math.Cos((2.0 \* j + 1) \* i \* (3.14 / (2.0 \* size)));

else

DCT[i, j] = 2.0 \* Math.Sqrt(size) \* Math.Cos((2.0 \* j + 1) \* i \* (3.14 / (2.0 \* size)));

label1.Text += String.Format("{0,-11}",Math.Round(DCT[i, j],6));

}

label1.Text += "\n";

}

label1.Text += "DCT-транспонированная:\n";

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

{

for (int j = 0; j < size; j++)

{

DCTT[i, j] = DCT[j, i];

label1.Text += String.Format("{0,-11} ", Math.Round(DCTT[i, j],6));

}

label1.Text += "\n";

}

double [,] PDCT1 = new double[size, size];

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

{

for (int j = 0; j < size; j++)

{

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

{

PDCT1[i, j] += DCT[i, k] \* P[k, j];

}

}

}

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

{

for (int j = 0; j < size; j++)

{

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

{

PDCT[i, j] += PDCT1[i, k] \* DCTT[k, j];

}

}

}

label1.Text += "PDCT:\n";

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

{

for (int j = 0; j < size; j++)

label1.Text += String.Format("{0,-16}", Math.Round(PDCT[i, j],4));

label1.Text += "\n";

}

}

private void button3\_Click(object sender, EventArgs e)

{

QUANT = new double[size, size];

PQDCT = new double[size, size];

double q = Convert.ToDouble(textBox1.Text);

label1.Text += "Матрица квантования:\n";

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

{

for (int j = 0; j < size; j++)

{

QUANT[i, j] = 1 + ((1 + i + j) \* q);

label1.Text += String.Format("{0,-5} ", QUANT[i, j]);

}

label1.Text += "\n";

}

label1.Text += "PQDCT:\n";

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

{

for (int j = 0; j < size; j++)

{

PQDCT[i, j] = PDCT[i,j]/QUANT[i,j];

label1.Text += String.Format("{0,-10} ", Math.Round(PQDCT[i, j]));

}

label1.Text += "\n";

}

}

private void button4\_Click(object sender, EventArgs e)

{

double[,] UP = new double[size, size];

double[,] UP1 = new double[size, size];

double[,] UP2 = new double[size, size];

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

{

for (int j = 0; j < size; j++)

{

UP1[i, j] = PQDCT[i, j] \* QUANT[i, j];

}

}

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

{

for (int j = 0; j < size; j++)

{

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

{

UP2[i, j] += Gauss(DCT)[i,k]\*UP1[k,j];

}

}

}

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

{

for (int j = 0; j < size; j++)

{

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

{

UP[i, j] += UP2[i, k] \* Gauss(DCTT)[k, j];

}

}

}

label1.Text += "Распакованная матрица:\n";

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

{

for (int j = 0; j < size; j++)

{

UP[i, j] = Math.Round(UP[i, j]) + 128;

label1.Text += String.Format("{0,-11} ", UP[i, j]);

}

label1.Text += "\n";

}

//Проверка на совпадение матриц

bool flag = false;

string C=null;

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

{

for (int j = 0; j < size; j++)

{

if (UP[i, j] == P[i, j]+128) flag = true;

else {flag = false; C+=i.ToString()+";"+j.ToString()+" ";}

}

}

if(flag==true)

label1.Text += "Матрицы совпадают.";

else label1.Text += "Матрицы на совпадают."+C;

}

static public double[,] Gauss (double [,] array)

{

double[,] aRevers = new double[size, size];

double[,] aCopy = new double[size, size];

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

{

for (int j = 0; j < size; j++)

{

if (i == j) aRevers[i, j] = 1;

else aRevers[i, j] = 0;

aCopy[i, j] = array[i, j];

}

}

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

{

double div = aCopy[k, k];

for (int m = 0; m < size; m++)

{

aCopy[k, m] /= div;

aRevers[k, m] /= div;

}

for (int i = k + 1; i < size; i++)

{

double multi = aCopy[i, k];

for (int j = 0; j < size; j++)

{

aCopy[i, j] -= multi \* aCopy[k, j];

aRevers[i, j] -= multi \* aRevers[k, j];

}

}

}

for (int k = size - 1; k > 0; k--)

{

aCopy[k, size - 1] /= aCopy[k, k];

aRevers[k, size - 1] /= aCopy[k, k];

for (int i = k - 1; i + 1 > 0; i--)

{

double multi = aCopy[i, k];

for (int j = 0; j < size; j++)

{

aCopy[i, j] -= multi \* aCopy[k, j];

aRevers[i, j] -= multi \* aRevers[k, j];

}

}

}

return aRevers;

}

}

}