**Lab-13**

**Name : Divan MunafSha Salimsha**

**Roll No : CE035**

**Subject : NIS**

**Student ID: 19CEUBG006**

**Aim: Write a program to implement Steganography using 1,2 and 3-LSB. (you can specify bits)**

* **Embedding**
* **Extraction**
* **MSE**
* **PSNR**

**Source Code:**

**Programming Language : C++**

**Ans:**

**Notes for running : M is the Message size you need to change it and change bits used accordingly for correct answer.**

#include "../functions.h"

#define N 8

#define M 192

#define Image vector<vector<unsigned long>>

using namespace std;

double mean\_squared\_error(Image img1, Image img2)

{

    double mse = 0;

    for (int i = 0; i < N; ++i)

    {

        for (int j = 0; j < N; ++j)

        {

            int m = img1[i][j];

            int n = img2[i][j];

            int diff = abs(int(m - n));

            mse += (diff \* diff \* 1.0 / (N \* N));

        }

    }

    cout << fixed << "mean squared error: " << mse << endl;

    return mse;

}

string extract(Image stego, int bit)

{

    bitset<M> msg(0);

    int k = 0;

    for (int i = 0; i < N; ++i)

    {

        for (int j = 0; j < N; ++j)

        {

            bitset<M> pixel(stego[i][j]);

            for (int l = 0; l < bit; ++l)

                msg[k++] = pixel[l];

            if (k >= msg.size())

                break;

        }

        if (k >= msg.size())

            break;

    }

    return msg.to\_string();

}

Image embed(string msg, Image image, int bit)

{

    bitset<2> value(1);

    int k = 0;

    for (int i = 0; i < N; ++i)

    {

        for (int j = 0; j < N; ++j)

        {

            bitset<M> pixel(image[i][j]);

            for (int l = 0; l < bit; ++l)

            {

                if (msg[k++] == '1')

                    pixel[l] = value[0];

                else

                    pixel[l] = value[1];

            }

            image[i][j] = pixel.to\_ulong();

            if (k >= msg.length())

                break;

            // cout<stego[i][j]<<endl;

        }

        if (k >= msg.size())

            break;

    }

    return image;

}

void peak\_signal\_to\_noise\_ratio(Image img1, Image img2, double mse)

{

    double psnr = 10 \* log10(255 \* 255.0 / mse);

    cout << "Peak signal to Noise Ratio: " << psnr << endl;

}

int main()

{

    ifstream input("Input.txt");

    Image image(N, vector<unsigned long>(N));

    for (int i = 0; i < N; ++i)

        for (int j = 0; j < N; ++j)

            input >> image[i][j];

    string msg;

    input >> msg;

    int bit;

    input >> bit;

    cout << bit << endl;

    ofstream output("Output.txt");

    Image stego\_image = embed(msg, image, bit);

    cout << "\n-------------------------Original Image--------------------------------" << endl;

    for (int i = 0; i < N; ++i)

    {

        for (int j = 0; j < N; ++j)

        {

            cout << image[i][j] << " ";

        }

        cout << endl;

    }

    cout << "-------------------------Stego Image--------------------------------" << endl;

    for (int i = 0; i < N; ++i)

    {

        for (int j = 0; j < N; ++j)

        {

            cout << stego\_image[i][j] << " ";

            output << stego\_image[i][j] << " ";

        }

        output << endl;

        cout << endl;

    }

    string extracted = extract(stego\_image, bit);

    cout << "Used bits: " << bit << endl;

    cout << "Original Msg: " << msg << endl;

    cout << "Extracted Msg: " << extracted << endl;

    double mse = mean\_squared\_error(image, stego\_image);

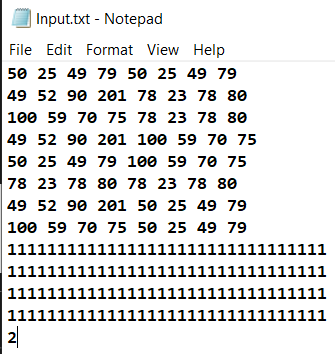
peak\_signal\_to\_noise\_ratio(image, stego\_image, mse);

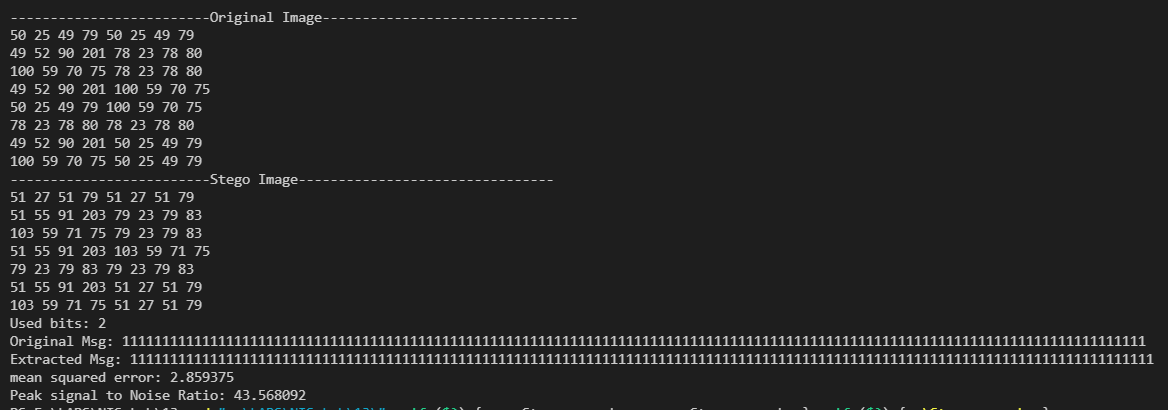
    return 0;

}

**Input & Output Screenshots:**

* ***For 2 bits and 128 bit binary message***





* ***For 3 bit and 192 bit binary message***

