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.

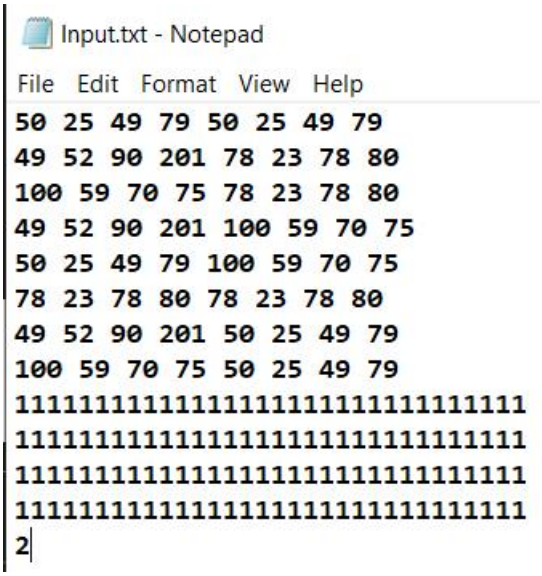
```
#include "../functions.h"
#define N 8
#define M 192
#define Image vector<vector<unsigned long>>
using namespace std;

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***




```
-------------------------Original Image--------------------------------
50 25 49 79 50 25 49 79
49 52 90 201 78 23 78 80
100 59 70 75 78 23 78 80
49 52 90 201 100 59 70 75
50 25 49 79 100 59 70 75
78 23 78 80 78 23 78 80
49 52 90 201 50 25 49 79
100 59 70 75 50 25 49 79
-------------------------Stego Image--------------------------------
51 27 51 79 51 27 51 79
51 55 91 203 79 23 79 83
103 59 71 75 79 23 79 83
51 55 91 203 103 59 71 75
79 23 79 83 79 23 79 83
51 55 91 203 51 27 51 79
103 59 71 75 51 27 51 79
Used bits: 2
Original Msg: 11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
Extracted Msg: 11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
mean squared error: 2.859375
Peak signal to Noise Ratio: 43.568092
```

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

Input.txt - Notepad

File Edit Format View Help

```
50 25 49 79 50 25 49 79
49 52 90 201 78 23 78 80
100 59 70 75 78 23 78 80
49 52 90 201 100 59 70 75
50 25 49 79 100 59 70 75
78 23 78 80 78 23 78 80
49 52 90 201 50 25 49 79
100 59 70 75 50 25 49 79
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
11111111111111111111111111111111
3
```

```
------------------------Original Image-------------------------------
50 25 49 79 50 25 49 79
49 52 90 201 78 23 78 80
100 59 70 75 78 23 78 80
49 52 90 201 100 59 70 75
50 25 49 79 100 59 70 75
78 23 78 80 78 23 78 80
49 52 90 201 50 25 49 79
100 59 70 75 50 25 49 79
------------------------Stego Image-------------------------------
55 31 55 79 55 31 55 79
55 55 95 207 79 23 79 87
103 63 71 79 79 23 79 87
55 55 95 207 103 63 71 79
55 31 55 79 103 63 71 79
79 23 79 87 79 23 79 87
55 55 95 207 55 31 55 79
103 63 71 79 55 31 55 79
Used bits: 3
Original Msg: 111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111
Extracted Msg: 11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
111111111111111111111111111111111111111111111111
mean squared error: 18.359375
Peak signal to Noise Ratio: 35.492225
PS E:\LABS\NIS Lab\13>
```