Lab-13

Name: Jenil J Gandhi

Roll-No: CE047

Subject: Network and Information Security.

Q-1)

Aim: Write a program to demonstrate Image Steganography operations: Embed and Extract

Hide 2 bits per pixel. Hide 3 bits per pixel.

Compute MSE (Mean Squared Error) and PSNR (Peak Signal to Noise Ratio) values.

Ans:

Code:

```
/*****************************************************
Written by:  Jenil J Gandhi
Subject:     Network and information security
Lab-no:      Lab-13
Description: Image 2 Bit Stegnography
Guidance by: Prof Mrudang T Mehta
Date:        28/03/2022
*****************************************************/
#include <bits/stdc++.h>
using namespace std;
int IMG_SIZE = 4;

void print_vec(vector<int> v)
```

```
{
    for (auto x : v)
    {
        cout << x;
    }
    cout << endl;
}

int getNumberFromBinaryVec(vector<int> binVec)
{
    reverse(binVec.begin(), binVec.end());
    int n = 0;
    for (int i = 0; i < binVec.size(); i++)
    {
        if (binVec[i] == 1)
        {
            n += (int(pow(2, float(i))));
        }
    }
    return n;
}

vector<int> getBinVec(int num, int pad)
{
    vector<int> bin;
    int c = 0;
    while (num > 0)
    {
        bin.insert(bin.begin(), num % 2);
        num /= 2;
        c++;
    }
    int padding = pad - c;
    if (padding > 0)
    {

        for (int i = 0; i < padding; i++)
        {
            bin.insert(bin.begin(), 0);
        }
    }
```

```
    return bin;
}

vector<int> getBinaryVector(int num, vector<int> number)
{
    vector<int> bin;
    while (num > 0)
    {
        bin.insert(bin.begin(), num % 2);
        num /= 2;
    }

    int cntr = 1;
    int size = bin.size();
    reverse(number.begin(), number.end());

    for (int i = 0; i < number.size(); i++)
    {
        bin[size - cntr++] = number[i];
    }
    // print_vec(bin);
    return bin;
}

vector<vector<int>> getSteagnoObj(vector<vector<int>> image, vector<int>
message, int pad)
{
    int c = 0;
    vector<vector<int>> stegoImage;
    for (auto x : image)
    {
        vector<int> row;
        for (auto pixel : x)
        {
            int msg = message[c++];
            vector<int> binaryMessage = getBinVec(msg, pad);
            vector<int> binaryImagePixel = getBinaryVector(pixel,
binaryMessage);
            int overlappedPixel =
getNumberFromBinaryVec(binaryImagePixel);
            row.push_back(overlappedPixel);
```

```
        }
        stegoImage.push_back(row);
    }
    return stegoImage;
}

vector<int> getDecryptedMessage(vector<vector<int>> stegoImage, int num)
{
    vector<int> resultVec;
    for (auto row : stegoImage)
    {
        for (auto pixel : row)
        {
            vector<int> binPixel = getBinVec(pixel, 0);
            reverse(binPixel.begin(), binPixel.end());
            vector<int> resVec;
            for (int i = 0; i < num; i++)
            {
                resVec.push_back(binPixel[i]);
            }
            reverse(resVec.begin(), resVec.end());
            int no = getNumberFromBinaryVec(resVec);
            resultVec.push_back(no);
        }
    }
    return resultVec;
}

double getLoss(vector<vector<int>> image, vector<vector<int>> stegoImage)
{
    double loss = 0;
    for (int i = 0; i < IMG_SIZE; i++)
    {
        for (int j = 0; j < IMG_SIZE; j++)
        {
            loss += ((image[i][j] - stegoImage[i][j]) * (image[i][j] -
stegoImage[i][j]));
        }
    }
    loss /= (16);
    return loss;
```

```
}

double getPSNR(int r, double MSE)
{
    return 10 * log10(double(r * r) / MSE);
}

void print__vec(vector<int> v)
{
    for (auto x : v)
    {
        cout << x << " ";
    }
    cout << endl;
}

int main()
{
    int tt;
    cin >> tt;
    while (tt--)
    {
        int bitDecr;
        cin >> bitDecr;
        vector<vector<int>> image{{50, 25, 49, 79}, {78, 23, 78, 80}, {49,
52, 90, 201}, {100, 59, 70, 75}};
        vector<int> message;
        for (int i = 0; i < 16; i++)
        {
            int tmp;
            cin >> tmp;
            message.push_back(tmp);
        }
        vector<vector<int>> stegoImage = getSteagnoObj(image, message,
bitDecr);
        vector<int> decryptedMessage = getDecryptedMessage(stegoImage,
bitDecr);
        cout << "> Decrypted Message\n";
        print__vec(decryptedMessage);
        cout << "> Augmented Image\n";
        for (auto x : stegoImage)
```

```
        {
            for (auto y : x)
            {
                cout << y << " ";
            }
            cout << endl;
        }
        cout << "> Loss:" << getLoss(image, stegoImage) << endl;
        cout << "> PSNR:" << getPSNR(4, getLoss(image, stegoImage)) <<
endl;
    }
    return 0;
}
```

Examples of Input and Output:

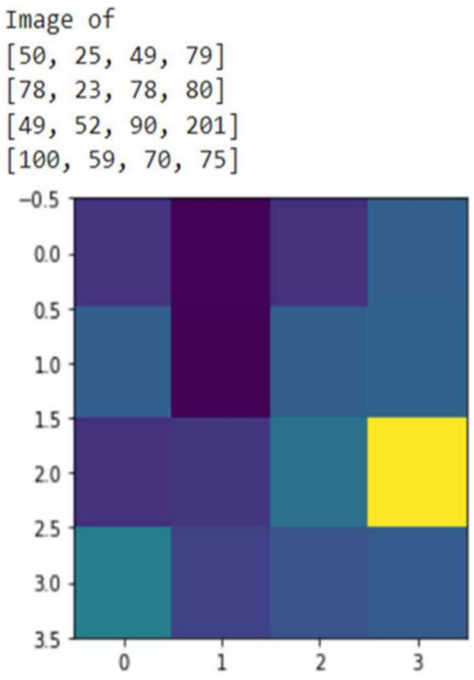
Input.txt

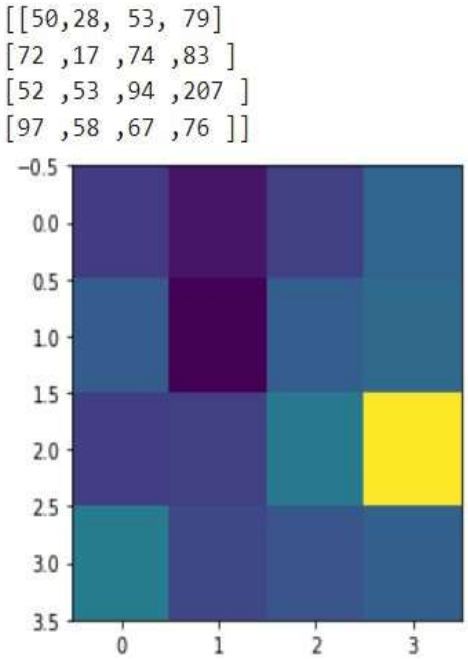
```
2
2
0 2 3 2
2 3 1 1
0 1 1 1
1 0 0 0

3
2 4 5 7
0 1 2 3
4 5 6 7
1 2 3 4
```

Output.txt

```
--------------------------------------------------------
Case #1
> Decrypted Message
0 2 3 2 2 3 1 1 0 1 1 1 1 0 0 0
```

```
> Augmented Image
48 26 51 78
78 23 77 81
48 53 89 201
101 56 68 72
> Loss:2.375
> PSNR:8.28456
------------------------------------------------------------
Case #2
> Decrypted Message
2 4 5 7 0 1 2 3 4 5 6 7 1 2 3 4
> Augmented Image
50 28 53 79
72 17 74 83
52 53 94 207
97 58 67 76
> Loss:12.75
> PSNR:0.986098
------------------------------------------------------------
```





Original Image **Stego Image**

For Testcase-2