Course: CS 825
Name: Chao Zhang
Student#: 200383834
Programming language: JAVA

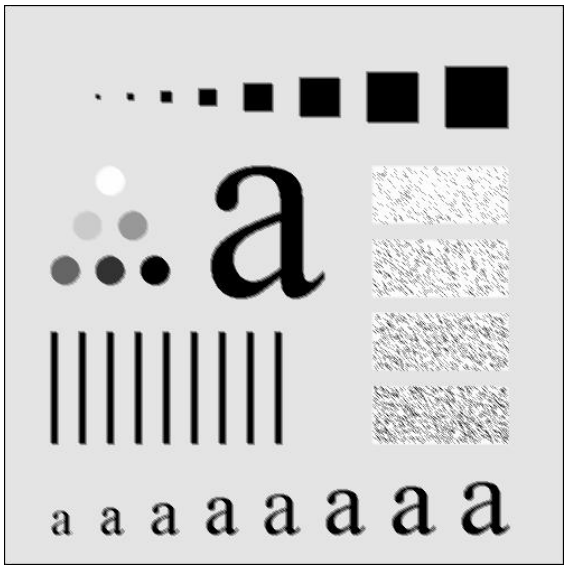
Programming 1

```
package Programming;

import java.io.*;

public class P1 {

	public static void main(String args[]) throws IOException {
		FileInputStream in = null;
		FileOutputStream out = null;
		try {
			in = new FileInputStream("D:/ct.raw");
			out = new FileOutputStream("D:/ct+.raw");
			int i = 0, j = 0;
			int[][] image_in = new int[256][256];

			for (i = 0; i < 256; i++)
				for (j = 0; j < 256; j++)
					image_in[i][j] = in.read(); //Read the input image into
image_in[ ][ ]

			int[][] image_out = new int[256][256];
```

```
                int[] h = new int[256];
                int[] H = new int[256];

                for (i = 0; i < 256; i++)    // initialization of h[]
                    h[i] = 0;

                for (i = 0; i < 256; i++)
                    for (j = 0; j < 256; j++)
                        h[image_in[i][j]]++;      // Compute the histogram of the input
image and store it in h[ ]

                H[0] = h[0];
                for (i = 1; i < 256; i++)      //Compute the cumulative histogram and store it
in H[]
                    H[i] = H[i - 1] + h[i];

                double s = 0.00389;    //get the scaling factor S
                //0.00389 is k-1/m*n which is 255(8 bits grayscale)/256*256

                for (i = 0; i < 256; i++) //Normalize H[] with the scaling factor S
                    H[i] *= s;

                for (i = 0; i < 256; i++)
                    for (j = 0; j < 256; j++)      //get the image_out[] from the H[]
                        image_out[i][j] = H[image_in[i][j]];

                for (i = 0; i < 256; i++)
                    for (j = 0; j < 256; j++)
                        out.write(image_out[i][j]); //Write the result image image_out[ ][ ]

            } finally {
                if (in != null) {
                    in.close();
                }
                if (out != null) {
                    out.close();
                }
            }

        }
}
```

Programming 2

(1) Using filter h1:



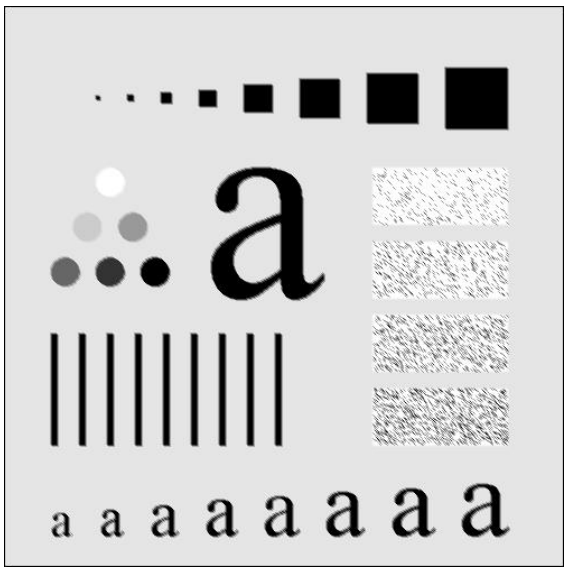
```
package Programming;

import java.io.*;

public class P2 {

    public static void main(String args[]) throws IOException {
        FileInputStream in = null;
        FileOutputStream out = null;
        try {
            in = new FileInputStream("D:/testpattern.raw");
            out = new FileOutputStream("D:/testpattern+.raw");
            int i, j;
            int[][] image_in = new int[500][500];
            int[][] image_out = new int[500][500];

            for (i = 0; i < 500; i++)
                for (j = 0; j < 500; j++)
                    image_in[i][j] = in.read(); // Read the input image into
image_in[ ][ ]
```

```
            //h1 is the first filter matrix, h2 is the second filter matrix, which are the two
3*3 filters
            double[][] h1 = { { 0.111, 0.111, 0.111 }, { 0.111, 0.111, 0.111 }, { 0.111,
0.111, 0.111 } };
            double[][] h2 = { { 0.075, 0.125, 0.075 }, { 0.125, 0.200, 0.125 }, { 0.075,
0.125, 0.075 } };

            int k1, k2;
            double sum;
            for (i = 1; i < 499; i++)    // i and j from 1 to 498 is to ignore some pixels in
four corners because these pixels can not be proceed by filter
                for (j = 1; j < 499; j++) {
                    sum = 0;       //initialization of sum for every pixel
                    for (k1 = -1; k1 <= 1; k1++)
                        for (k2 = -1; k2 <= 1; k2++)
                            sum += image_in[i + k1][j + k1] * h1[k1 + 1][k2 + 1];
//use filter matrix h1 or h2 to filter the image respectively

                    image_out[i][j] = (int) sum;
                }

            for (i = 0; i < 500; i++)
                for (j = 0; j < 500; j++)
                    out.write(image_out[i][j]);    //Write the result image
image_out[ ][ ]

        } finally {
            if (in != null) {
                in.close();
            }
            if (out != null) {
                out.close();
            }
        }

    }
}
```

(2) Using filter h2

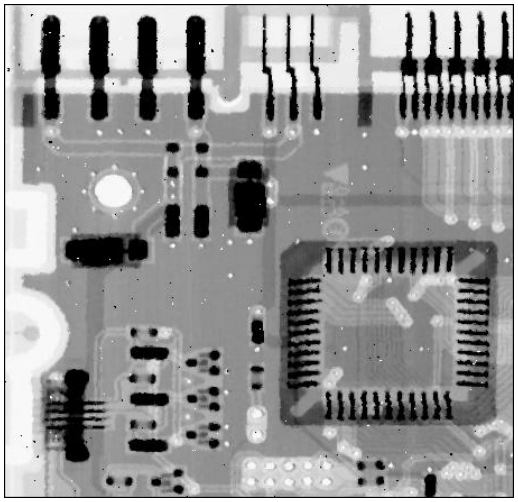
Change the filter matrix h1 to h2

```
int k1, k2;
double sum;
for (i = 1; i < 499; i++)
    for (j = 1; j < 499; j++) {
        sum = 0;    //initialization of sum for every pixel
        for (k1 = -1; k1 <= 1; k1++)
            for (k2 = -1; k2 <= 1; k2++)
                sum += image_in[i + k1][j + k1] * h2[k1 + 1][k2 + 1];
        image_out[i][j] = (int) sum;
    }
```



Conclusion: We can not see visual difference between the two results.

Programming 3



```
package Programming;

import java.io.*;

public class P3 {

    public static void main(String args[]) throws IOException {
        FileInputStream in = null;
        FileOutputStream out = null;
        try {
            in = new FileInputStream("D:/circuit.raw");
            out = new FileOutputStream("D:/circuit+.raw");
            int i, j;
            int[][] image_in = new int[440][455];
            int[][] image_out = new int[440][455];

            for (i = 0; i < 440; i++)
                for (j = 0; j < 455; j++)
                    image_in[i][j] = in.read(); // Read the input image into
                                                // image_in[][]

            int k1, k2, counter;
            int[] mid = new int[9];

            for (i = 1; i < 439; i++) // i from 1 to 438 and j from 1 to 453 is
                                      // to ignore some pixels in four corners
```

```
                                        // because these pixels can not be
                                        // proceed by filter as boundaries
                for (j = 1; j < 454; j++) {

                    counter = 0;
                    for (k1 = -1; k1 <= 1; k1++)
                        for (k2 = -1; k2 <= 1; k2++)
                            mid[counter++] = image_in[i + k1][j + k2];
                    // put the filter's 9 pixels in mid[] to sort

                    image_out[i][j] = sort(mid); // class sort is to sort the
                                                 // mid[] and return the
                                                 // median number of these 9
                                                 // pixels in the filter
                }

            for (i = 0; i < 440; i++)
                for (j = 0; j < 455; j++)
                    out.write(image_out[i][j]); // Write the result image
                                                // image_out[][]
        } finally {
            if (in != null) {
                in.close();
            }
            if (out != null) {
                out.close();
            }
        }

}

public static int sort(int[] mid) { // bubble sort
    int sign = 1, t, kk;
    while (sign != 0) {
        sign = 0;
        for (kk = 0; kk < 8; kk++)
            if (mid[kk] > mid[kk + 1]) {
                t = mid[kk + 1];
                mid[kk + 1] = mid[kk];
                mid[kk] = t;
                sign = 1;
```

```
                    }
                }

                return mid[4]; // return the median number
            }
        }
```