```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Diverses
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function result = normalize01(x)
        minval = min(min(x));
        maxval = max(max(x));
        scale = maxval - minval
        x = x - minval;
        x = x / (maxval - minval);
        result = x;
end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aufgabe 2 a)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function w = einheitswurzel(n, k)
        w = e^(i*2*pi/n*k);
end;

function A = df(n) % Erstellen der Fourier-Matrix
        A = complex(zeros(n));
        for x = 1 : size(A, 1)
                for y = 1 : size(A, 2)
                        A(x, y) = 1/sqrt(n)*einheitswurzel(n, x * y);
                end
        end
end;

%%%%% Erstellen und speichern der 512-er Fourier-Matrix
%f = df(512)
%save("-binary","f512","f")

%%%% Die 512er Fourier-Matrix laden
load("f512");
global f512 = f;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aufgabe 2 b)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function A = fourier(B) % Transformation
        global f512;
        A = f512 * complex(B) * f512';
end;

function run(img, basename)
        global f512;
        SPEKTRUM = f512*complex(img)*(f512)';
        imwrite(uint8(real(SPEKTRUM) * 255), strcat(basename, ".real.png"));
        imwrite(uint8(imag(SPEKTRUM) * 255), strcat(basename, ".image.png"));
        imwrite(uint8(abs(SPEKTRUM) * 255), strcat(basename, ".length.png"));
        imwrite(uint8(angle(SPEKTRUM) * 255), strcat(basename, ".angle.png"));
end;

%SQ1 = imread("square1.png");
%SQ2 = imread("square2.png");
%SQ3 = imread("square3.png");
%SQ4 = imread("square4.png");
%
%run(SQ1, "square1")
%run(SQ2, "square2")
%run(SQ3, "square3")
%run(SQ4, "square4")

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aufgabe 2 c)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function A = unfourier(B) % Rücktransformation
        global f512;
        A = uint8(real(conj(f512) * B * conj(f512')));
end;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aufgabe 2 d)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

LENA = imread("lena-bw.png");
LENA_SPEKTRUM = fourier(LENA);

%%%%% Testen der Hin- und Rücktransformation mit Lena
%imshow(unfourier(fourier(LENA)), [0, 255])

function val = percentile(X, p)
        % return the value that p * length(X)
        % values of X are smaller or equal to.
        S = sort(X);
        len = size(S, 1);
        pos = round(p * len) + 1;
        percent = pos / len
        val = S(pos)
end;

%imshow(unfourier(LENA_SPEKTRUM))
%imshow(unfourier(LENA_MOD))

function A = cutLowAbs(B, p)
        A = B;
        S = abs(A);
        cut = percentile(S(:), p);
        A(find(S < cut)) = 0;
end;

function A = cutLowImag(B, p)
        A = B;
        S = imag(A);
        cut = percentile(S(:), p);
        A(find(S < cut)) = 0;
end;

function A = cutLowReal(B, p)
        A = B;
        S = real(A);
        cut = percentile(S(:), p);
        A(find(S < cut)) = 0;
end;

function A = cutLowAngle(B, p)
        A = B;
        S = angle(A);
        cut = percentile(S(:), p);
        A(find(S < cut)) = 0;
end;

for p = [0.0001 0.001 0.01 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 0.99 0.999 0.9999]
        name = sprintf("lena-low-abs-%f.png", p);
        printf("%s\n", name);
        imwrite(unfourier(cutLowAbs(LENA_SPEKTRUM, p)), name);
end;

for p = [0.0001 0.001 0.01 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 0.99 0.999 0.9999]
        name = sprintf("lena-low-angle-%f.png", p);
        printf("%s\n", name);
        imwrite(unfourier(cutLowAngle(LENA_SPEKTRUM, p)), name);
end;

for p = [0.0001 0.001 0.01 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 0.99 0.999 0.9999]
        name = sprintf("lena-low-real-%f.png", p);
        printf("%s\n", name);
        imwrite(unfourier(cutLowReal(LENA_SPEKTRUM, p)), name);
end;

for p = [0.0001 0.001 0.01 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 0.99 0.999 0.9999]
        name = sprintf("lena-low-imag-%f.png", p);
        printf("%s\n", name);
        imwrite(unfourier(cutLowImag(LENA_SPEKTRUM, p)), name);
end;
```