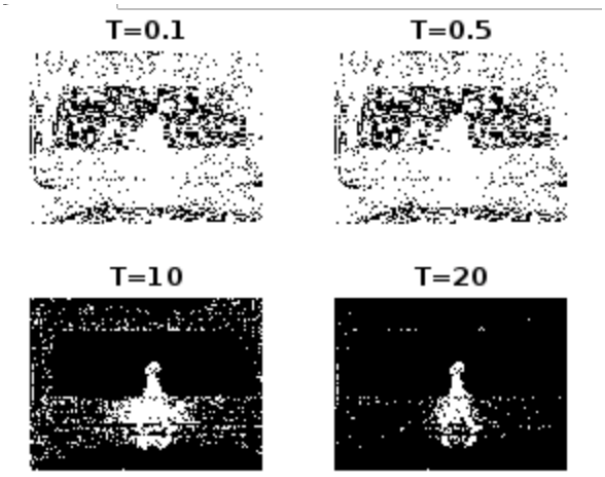
Code:

```
%% Problem 1
myRbgIm = imread('im.png');
myIm = double(im2gray(myRbgIm));
a = 0.4;
N = 3;

g1 = blur(myIm,0.4);
g1 = reduce(g1);
subplot(2,3,1);
imshow(g1);
g2 = blur(g1, 0.4);
g2 = reduce(g2);
subplot(2,3,2);
imshow(g2);
g3 = blur(myIm, 0.4);
g3 = reduce(g3);
subplot(2,3,3);
imshow(g3);

l3 = g3;
l2 = expand(g2) - g1;
l1 = expand(g1) - myIm;

recontruction = l1 + expand(l2) + expand(l3);
imshow(recontruction);

% function
function image_blured = blur(img,a)
    x = [.25-.5.*a .25 a .25 .25-.5.*a];
    y = [.25-.5.*a; .25; a; .25; .25-.5.*a];
    %blur
    temp = imfilter(img, x);
    image_blured = imfilter(temp, y);
end


function img_reduced = reduce(img)
    [NR, NC] = size(img);
    img_reduced = zeros(uint8(NR/2), uint8(NC/2));


    for r = 1:NR/2
        for c = 1:NC/2
            r_indices = 2*r-1:2*r;
            c_indices = 2*c-1:2*c;
            img_reduced(r,c) = mean(mean(img(r_indices, c_indices)));
        end
    end
end

function img_expanded = expand(img)
    [NR, NC] = size(img);
    %disp(NR);
    %disp(NC);
    img_expanded = zeros(2*NR, 2*NC);

    img_expanded(1:2:end, 1:2:end) = img;
    img_expanded(2:2:end, 2:2:end) = img;

    for r = 2:2:2*NR-1
        for c = 2:2:2*NC-1
            img_expanded(r,c) = (img_expanded(r-1,c) + img_expanded(r+1,c))/2;
        end
    end
end
```

Here is what I did for Problem one. I meet some problems here and I spent at least 20 hours on this question but I still can't solve it. I designed three functions. The first one is blur(img, a) which is used to blur the image. The last two functions is used to resize.

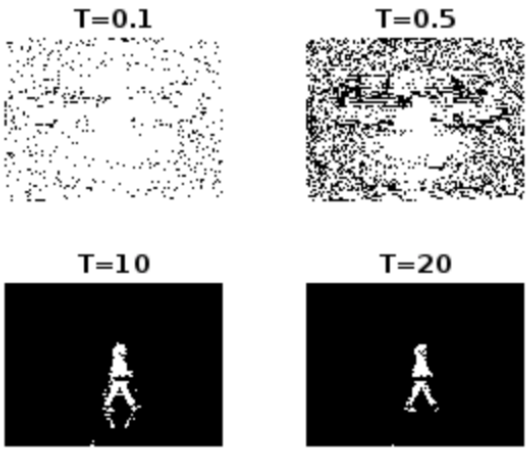
```
%% Problem 2
walk = imread('walk.bmp');
bg = imread('bg000.bmp');

walk = double(walk);
bg = double(bg);

diff = abs(walk - bg);

threshold = [0.1, 0.5,10,20];
i = 1;
for T = threshold
    im = diff > T;
    subplot(2,2,i);
    imshow(im);
    i = i+1;
    title("T=" +T);
end
```



Here is what did for question 2. I set T to 0.1, 0.5, 10, and 20. With a bigger Threshold, the output becomes clearer.

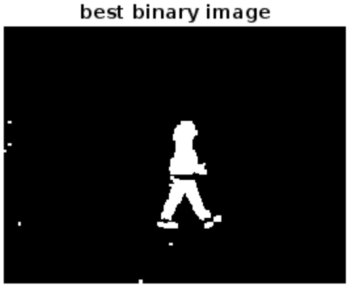
```
%% Problem 3
for i=1:30
filename = sprintf('bg%03d.bmp', i-1);
bg(:,:,i) = double(imread(filename));
end

walk = imread('walk.bmp');
walk = double(walk);

meanBg = mean(bg, 3);
%calculate the standard deviation
sig = zeros(240,320);
for j=1:30
sig = sig+((bg(:,:,j) - meanBg).^2)/30;
end
sig = sqrt(sig);

sDistance =abs(walk - meanBg)./sig;

threshold = [0.1, 0.5,10,20];
i = 1;
for T = threshold
    im = sDistance > T;
    subplot(2,2,i);
    imshow(im);
    i = i+1;
```

```
    title("T=" +T);
end
```



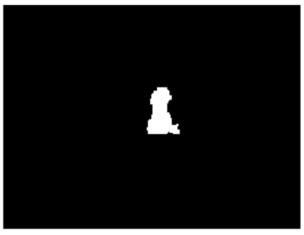
Here is what I did for problem 3. There are fewer noises in the output compared with problem 2.

```
%% problem 4
T =20;
bsIm = sDistance > T;
d_bsIm = bwmorph(bsIm, 'dilate');
imshow(double(d_bsIm));
imwrite(bsIm, 'bestBinary.jpg');
title("best binary image");
```



Here is what I did for Problem 4. I dilate my best binary image.

```
%% problem 5
[L, num] = bwlabel(d_bsIm, 8);
stats = regionprops(L, 'Area');
areas = [stats.Area];
[~, maxIndex] = max(areas);
largest = ismember(L, maxIndex);
imshow(largest);
imwrite(largest, 'largest.jpg');
```



Here is what I did for problem 5. I perform the connected components algorithm and keep the largest region in L.