Guidance, tips and fun for functions 7 and 8

Fun with function 7 (get_angle):

Let's have a fun examination of an example for function 7 together. Let's assume we have the image:

```
image = [[0  ,0  ,0  ,0  ,0  ,0  ,255],
         [0  ,0  ,0  ,0  ,0  ,255,0  ],
         [255,0  ,0  ,0  ,0  ,0  ,0  ],
         [0  ,0  ,0  ,255,0  ,0  ,0  ],
         [0  ,0  ,255,0  ,0  ,0  ,0  ],
         [0  ,255,0  ,0  ,0  ,0  ,0  ],
         [255,0  ,0  ,0  ,0  ,0  ,0  ]]
```

Now, let's assume we want to give a "score" for a line in angle 135 degrees, and distance 5 from the origin (the image[0][0] pixel). This line goes through each of the pixels in the (secondary) diagonal - which means we should get a high value for it!

By calling:

```
line = ex6_helper.pixels_on_line(image, math.pi*135/180, 5)
```

we get the coordinates of the actual pixels that this line intersects (up to some roundings), which are:

```
[6,0],[5,1],[4,1],[4,2],[3,2],[3,3],[2,3],[2,4],[1,4],[1,5],[0,5],[0,6]
```

which are the yellow pixels in the following matrix:

```
image = [[0  ,0  ,0  ,0  ,0  ,0  ,255],
         [0  ,0  ,0  ,0  ,0  ,255,0  ],
         [255,0  ,0  ,0  ,0  ,0  ,0  ],
         [0  ,0  ,0  ,255,0  ,0  ,0  ],
         [0  ,0  ,255,0  ,0  ,0  ,0  ],
         [0  ,255,0  ,0  ,0  ,0  ,0  ],
         [255,0  ,0  ,0  ,0  ,0  ,0  ]]
```

Now, in order to get the value of this line, we look at the first pixel on it (the one at [6,0]). This pixel is white, so we remember it as a beginning of a new line. We move to the next pixel (at [5,1]), and this one is also white! So we are still on the same white line – as we can tell by the fact that the distance between the pixel at [6,0] and the pixel at [5,1] is smaller than 2. Great.

Next we have the pixel at [4,1]. This pixel is black. Ok, so it's possible that the last white pixel (at [5,1]) was the end of the line – but we don't know it yet. We go on to the next pixel, at [4,2]. This pixel is again white, so we check its distance to the "last white pixel we found" – and see that they are closer than 2! So this new pixel, at [4,2], is still part of our line. We proceed until we get to the pixel at [3,3], which is still on our line. Now, the next time we encounter a white pixel it will be at [1,5]. The distance between this pixel and the one at [3,3] is sqrt($(3-1)^2+(5-3)^2$)=sqrt(8), which is of course greater than 2. So the previous line ended at [3,3]. We calculate it's length as the distance between its first white pixel and its last white pixel, meaning: sqrt($(6-3)^2+(0-3)^2$)=sqrt(18), and then add its square, 18, to the value of this line. We proceed with finding the length of the line which starts at the pixel [1,5], which continues until the end of the line – meaning that the last white pixel we found on it, at [0,6] is the last pixel of this second line. So the length of the second line is sqrt($(1-0)^2+(5-6)^2$)=sqrt(2) , and we add its square, 2, to the value of this line. So in total the value of this line is 20. Fun, isn't it?

Guidance for funtion 8 (rotate):
If we create our rotated image by going over every pixel in our (original) image and checking which pixel they are supposed to end up in, we will get that due to roundings, several pixels might end up in the same place, leaving black spots. Instead, the right way to approach the task would be to first calculate the size of the resulting picture – and create a black image of that size.
Then, we go over each of the pixels in the **result** image, and figure out which pixel it originated from (this is simple: if pixel1 moved to pixel2 by a rotation of α, then pixel2 moves to pixel 1 by a rotation of $-\alpha$).

One thing you will need to note, however, is that the equation you are given is for rotation about [0,0], while in the exercise you have to rotate about the center of the image!