\AM@currentdocname .png

.png

.png

\AM@currentdocname .png

Aufgabe 2

```

    # compute histograms
    img_histo = compute_cumHisto(img, 1)
```

```
def compute_cumHisto(img, binSize=1):

    return np.cumsum(bin_Histo(img, binSize))


def bin_Histo(img, bin=1):
    intervalls = np.ceil(256 / bin)
    histo = np.zeros(shape=intervalls)

    for x in range(0, img.width):
        for y in range(0, img.height):
```

\AM@currentdocname .png

.png

.png

```
\AM@currentdocname .png
```

Aufgabe 4

```
        index = (brightness * intervalls) / 256
        histo[index] += 1

    return histo


def match_Histo(img_histo, ref_histo):

    #img_histo . . . original histogram
    #ref_histo . . . reference histogram
    #returns the mapping function LUT to be applied to the image
```

Aufgabe 5.a) Wenn in einem Foto nur dunkle Bereich aufgehellt und helle Bereiche abgedunkelt werden können Details erhalten bleiben.

```python

    for i in range(0,255):
        P_i = img_histo[i] / img_histo[255]
        for j in range(0, 255):
            P_j = ref_histo[j] / ref_histo[255]
            if P_i == P_j :
                LUT[i] = j
                break;

    return LUT


def apply_LUT(img, lut):

    for x in range(0, img.width):
        for y in range(0, img.height):
            edit = img.getpixel((x, y))
            lut_edit = lut[edit].item()
            img.putpixel((x, y), lut_edit)

    return img



def rgb2gray(rgb):

    # convert to grayscale image (only one channel)
    return rgb.convert('L')


if __name__ == "__main__":

    # read img
    img = Image.open("bild01.jpg")
    ref = Image.open("bild02.jpg")

    # convert to grayscale
```