Name : Hafidz Ubaidillah
AU ID : 109712286
University Origin : UMG (Universitas Muhammadiyah Gresik)

Answer

1. +
   a. Image:. a visual representation of something.
   b. Digital image: an array or matrix of square pixels (picture elements) arranged in rows and columns.
   c. Digital image processing: processing digital image by means of a digital computer. By using computer algorithms to change contrast, reduce noise etc.
2. There are three types of digital image processing
   a. Low level processing
      - Digital image processing which involves simple operations such as noise reduction, contrast enhancement and image sharpening.
      - Input and output are images.

   b. Mid level processing
      - Digital image processing that involves segmentation (dividing a thing, an image, into several parts), describing objects to fit the desired digital processing and classifying / recognizing an object in an image.
      - Generally, the input used is an image and the output is an attribute of the image.

   c. High level processing
      - Digital image processing which involves the "make sense" element of an ensemble of object recognition, such as in image analysis, and at the far end of the continuum, performing the cognitive functions normally associated with vision .
3. +
   a. Pillow

```
pillow_image = Image.open('astronaut.png')

pillow_image.show()
# or
display(pillow_image)
```

- You have to import the ‘Image’ module from the ‘PIL’ library, so you can use the ‘open ()’ function to load an image.
- In line 3, ‘pillow_image’ variable has a ‘show ()’ function which is useful for displaying images stored in that variable. The image will appear in a new window
- In line 5, this function from the ‘IPython’ module. In practice in ‘jupyter notebook’, you don't need to import it again because it is automatically done by ‘jupyter notebook’. The image will appear below the line code box.

b. Scikit-Image

```
1 scikitImage_image = io.imread('astronaut.png')
2
3 io.imshow(scikitImage_image)
```

- You have to import the ‘io' module from the ‘skimage’ library, so you can use the ‘imread ()' function to load an image.
- In line 1, the ‘imread ()’ function of the ‘io’ module allows you to load images by filling in the parameters for the image directory you want.
- In line 3, using the 'imshow ()' function of the 'io' module by filling in the parameter with the 'variable name' (scikitImage_image). This function will display an image of that variable below the line of code box.

c. OpenCV

```
1 opencv_image = cv2.imread('astronaut.png')
2
3 cv2.imshow('image', opencv_image)
4 cv2.waitKey(0)
```

- You have to import the ‘cv2 ‘, so you can use the ‘imread ()’ function to load an image.
- In line 1, the ‘imread ()’ function of the ‘cv2’ module allows you to load images by filling in the parameters for the image directory you want.
- In line 3, , using the 'imshow ()' function of the 'cv2' module by filling in the parameter with a description and 'variable name' (opencv_image). This function will display an image of that variable below the line of code box.

- In line 4, I use this syntax so that the image can be displayed (no 'not responding' occurs) and the process will continue until the image is closed.

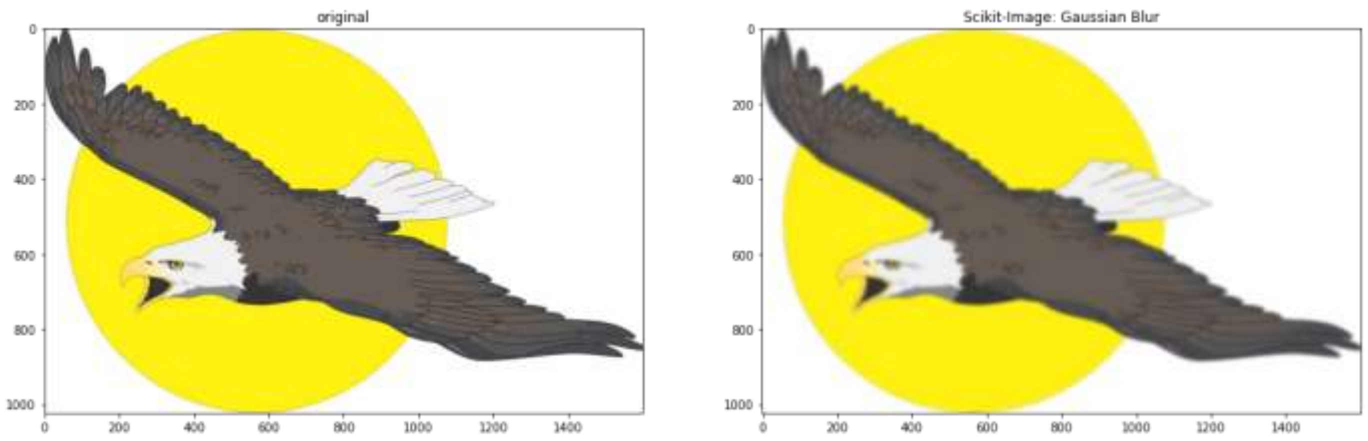
4. I've created a function to make it easier to work with the filters below and import the required modules.

```python
import matplotlib.pyplot as plt
from PIL import Image, ImageFilter, ImageEnhance
from skimage import io, data, filters, color, segmentation
from skimage.morphology import disk, dilation, erosion
from skimage.util import random_noise, img_as_ubyte
import numpy as np

def show_comparison(original, filtered, filtername, figsize=(20, 17)):
    fig, ax = plt.subplots(figsize=figsize, ncols=2)
    ax[0].imshow(original, cmap=plt.cm.gray)
    ax[0].set_title('original')
    ax[1].imshow(filtered, cmap=plt.cm.gray)
    ax[1].set_title(filtername)
```

a. Gaussian Blur

```python
ski_elangjawa_gau = filters.gaussian(elangjawa_ski, sigma=4)
show_comparison(elangjawa_ski, ski_elangjawa_gau, 'Scikit-Image: Gaussian Blur')

pil_elangjawa_gau = elangjawa_pil.filter(ImageFilter.GaussianBlur(4))
show_comparison(elangjawa_pil, pil_elangjawa_gau, 'Pillow: Gaussian Blur')
```

Output:



b. Median Filter

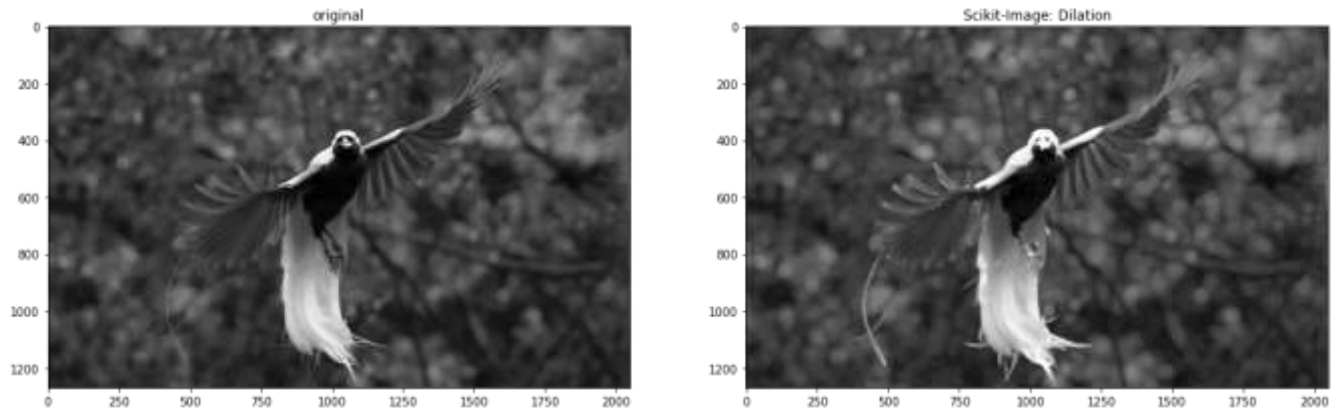
```
gray_cendrawasih = color.rgb2gray(cendrawasih_ski)

ski_cendrawasih_median = filters.rank.median(gray_cendrawasih,
                                             disk(5))
show_comparison(gray_cendrawasih,
                ski_cendrawasih_median,
                'Scikit-Image: Median Filter')

pil_cendrawasih_median = cendrawasih_pil.filter(ImageFilter.MedianFilter(7))
show_comparison(cendrawasih_pil,
                pil_cendrawasih_median,
                'Pillow: Median Filter')
```

Output:

c. Dilation and Erosion

```
dilated_ski = dilation(gray_cendrawasih,
                       disk(5))
show_comparison(gray_cendrawasih,
                dilated_ski,
                'Scikit-Image: Dilation')

eroded_ski = erosion(gray_cendrawasih
                     , disk(5))
show_comparison(gray_cendrawasih, eroded_ski,
                'Scikit-Image: Erosion')
```

Output:

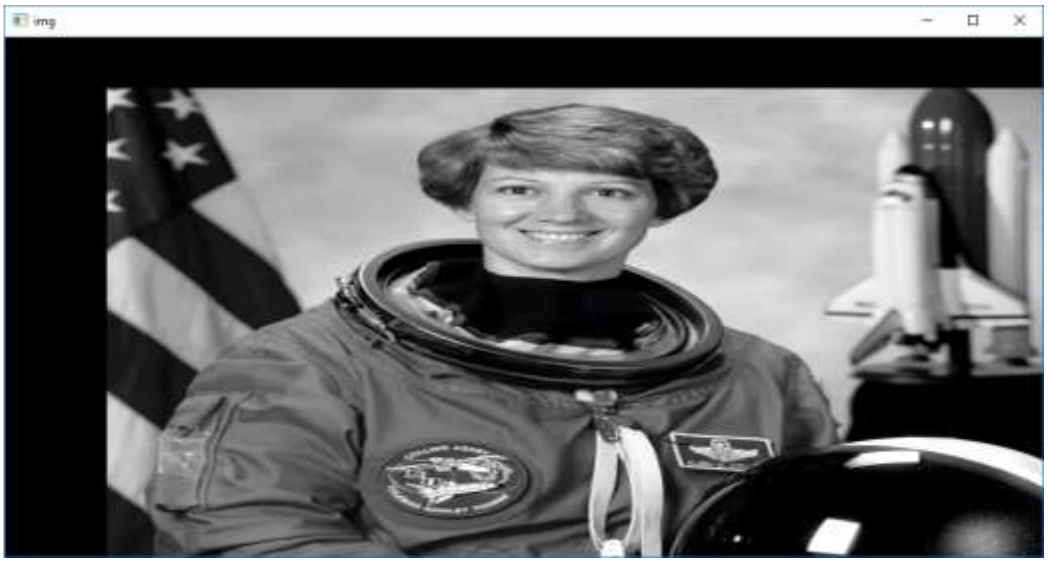


5. +

a. Transformation

```python
import numpy as np
import cv2 as cv
from skimage import color

# scaling
img = cv.imread('astronaut.png')
img = color.rgb2gray(img)
res = cv.resize(img,None,fx=2, fy=1,
                interpolation = cv.INTER_CUBIC)
#OR
height, width = img.shape[:2]
res = cv.resize(img,(2*width, 1*height),
                interpolation = cv.INTER_CUBIC)

# translation
rows,cols = res.shape
M = np.float32([[1,0,100],[0,1,50]])
dst = cv.warpAffine(res,M,(cols,rows))

cv.imshow('img',dst)
cv.waitKey(0)
```

Output:



b. Rotation

```python
import numpy as np
import cv2 as cv
from skimage import color

img = cv.imread('astronaut.png',0)
rows,cols = img.shape
# cols-1 and rows-1 are the coordinate limits.
M = cv.getRotationMatrix2D(((cols-1)/2.0,(rows-1)/2.0),90,1)
dst = cv.warpAffine(img,M,(cols,rows))

cv.imshow('img',dst)
cv.waitKey(0)
```

Output: