Lab 4

Applying filters on images.

| Filters types:

- Linear: (Mean - Gaussian) Linear filtering is the filtering method where the value of output pixel is linear combinations of the neighboring input pixels.
- Non-linear: (Min - Max - Median) Non-linear filter is a filter whose output is not a linear function of its input.

In [44]:
```
import cv2, numpy as np
```

In [45]:
```
image = cv2.imread('Pic3.jpg')
cv2.imshow('Original image', image)
```

| Non-Linear filter

Max Filter (Dilation Filter):

Applying morphological operation using a square-shaped structureing element

- It contains the regions of white pixels expanded or thickened.

_It is useful for joining broken parts of an object or increasing the size of foreground regions in an image.

_It works by moving the kernel over the image and replacing each pixel in the neighborhood with the maximum pixel value found in the kernel's corresponding region in the image.

In [46]:
```
kernel_size = 5
max_filtered_image = cv2.dilate(image, np.ones((kernel_size, kernel_size) , np.uint8))

cv2.imshow('Max Filtered Image', max_filtered_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- kernel size is 5 => the kernel is a square matrix with both dimensions equal to 5(5x5 matrix) : As each pixel in the output image is replaced by the median value of its 5x5 neighborhood.
- cv2.dilate() => function used to perform dilation.
- np.uint8 => used to ensure that the pixel values in the arrays are within the appropriate range for image data, and it also optimizes memory usage by using 8 bits per pixel.
- np.ones((kernel_size, kernel_size),np.uint8) => creates a kernel of size 5x5 filled with ones, which means it's a square structuring element of size 5x5.

Min Filter (Erosion Filter):

It is a morphological operation that shrinks regions of foreground pixels (usually white) in a binary image.

- It contains the regions of white pixels reduced or thinned

_It is useful for tasks like removing small noisy elements or disconnecting connected regions in an image.

_It works by replacing each pixel in the neighborhood with the minimum pixel value found in the kernel's corresponding region in the image.

In [47]:
```
kernel_size = 3
output_image = cv2.erode(image, np.ones((kernel_size, kernel_size), np.uint8))

#cv2.imshow('Input Image', image)
cv2.imshow('Min Filtered Image',output_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- Applying kernel size 3 to reducing the pixel of the image.
- cv2.erode(imageName, np.ones(,),) => performs the erosion operation on the image.

Median Filter:

_It is useful for removing noise from an image while preserving edges.

_It works by replacing each pixel value in the image with the median value of the pixel value in the image

In [52]:
```
kernel_size = 5

median_filtered_image = cv2.medianBlur(image, kernel_size)

cv2.imshow('Median Filtered Image', median_filtered_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- cv2.medianBlur() => function replaces each pixel value with the median value of the pixel values in the neighborhood.

| Linear Filter

Mean Filter(Box Filter):

_It is useful in bluring the image(smoothing an image and reducing noise).

_It works by replacing each pixel value in the image with the average (mean) value of the pixel values in its neighborhood defined by the kernel size.

In [49]:
```
kernel_size = (5, 5)

mean_filtered_image = cv2.blur(image, kernel_size) #The function arranges the pixels to determine

cv2.imshow('Mean Filtered Image', mean_filtered_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- cv2.blur(image, kernel_size) => applies median filtering operation to image.

Gaussian Blur:

_It is useful for bluring image.

_It applies a weighted aveg=rage to each pixel in the image(The weights defined by a Gaussian function).

In [50]:

```
kernel_size = (5,5)
blur = cv2.GaussianBlur(image, kernel_size, 0) #0 for standad deviation

cv2.imshow('Gaussian Blurred Image', blur)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- cv2.GaussianBlur(image, (5, 5), 0) => Applies the Gaussian blur filtering operation to the input image

---