Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering and Technology** **Department of Information and Communication Technology**	
Subject: DSIP (01CT1513)	**AIM: Smoothing and Sharpening in spatial domain**	
Experiment No: 09	**Date:**	**Enrolment No: 92301733046**

write a python program to simulate smoothening and sharpening operation on image using spatial filter

```
import cv2
import numpy as np
from google.colab.patches import cv2_imshow

# Load the image
image = cv2.imread('/content/smoothening data.png')

# Define the Gaussian kernel for smoothing
kernel_size = (5, 5)
sigma = 1.5
gaussian_kernel = cv2.getGaussianKernel(kernel_size[0], sigma)
gaussian_kernel = np.outer(gaussian_kernel, gaussian_kernel)

# Apply Gaussian smoothing
smoothed_image = cv2.filter2D(image, -1, gaussian_kernel)

# Define a sharpening kernel
sharpening_kernel = np.array([[-1, -1, -1],
                              [-1, 9, -1],
                              [-1, -1, -1]])

# Apply sharpening
sharpened_image = cv2.filter2D(image, -1, sharpening_kernel)

# Display the original image, smoothed, and sharpened images
cv2_imshow(image)
cv2_imshow(smoothed_image)
cv2_imshow(sharpened_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering and Technology** **Department of Information and Communication Technology**	
Subject: DSIP (01CT1513)	**AIM: Smoothing and Sharpening in spatial domain**	
Experiment No: 09	**Date:**	**Enrolment No: 92301733046**

Output:

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering and Technology** **Department of Information and Communication Technology**	
Subject: DSIP (01CT1513)	**AIM: Smoothing and Sharpening in spatial domain**	
Experiment No: 09	**Date:**	**Enrolment No: 92301733046**

write a python program to simulate smoothening using Averaging linear filters

```
import cv2
import numpy as np
from google.colab.patches import cv2_imshow
# Load the image
image = cv2.imread('/content/smoothening data.png')

# Define the size of the Averaging filter kernel
kernel_size = (5, 5)  # You can adjust the size based on the desired
smoothing level

# Create the Averaging filter kernel
kernel = np.ones(kernel_size, dtype=np.float32) / (kernel_size[0] *
kernel_size[1])

# Apply the Averaging filter for smoothing
smoothed_image = cv2.filter2D(image, -1, kernel)

# Display the original and smoothed images
cv2_imshow(image)
cv2_imshow(smoothed_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

output:

Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology		
Subject: DSIP (01CT1513)	AIM: Smoothing and Sharpening in spatial domain	
Experiment No: 09	Date:	Enrolment No: 92301733046

write a python program to simulate smoothening using median filters

```
import cv2
import numpy as np
from google.colab.patches import cv2_imshow

# Load the image
image = cv2.imread('/content/smoothening data.png')

# Define the size of the median filter kernel (should be an odd number)
kernel_size = 5  # You can adjust the size

# Apply the Median filter for smoothing
smoothed_image = cv2.medianBlur(image, kernel_size)

# Display the original and smoothed images using cv2_imshow
cv2_imshow(image)
cv2_imshow(smoothed_image)
```

Output:

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering and Technology** **Department of Information and Communication Technology**	
Subject: DSIP (01CT1513)	**AIM: Smoothing and Sharpening in spatial domain**	
Experiment No: 09	**Date:**	**Enrolment No: 92301733046**

write a python program to simulate sharpening using spatial high pass filters

```
import cv2
import numpy as np
from google.colab.patches import cv2_imshow
# Load the image
image = cv2.imread('/content/smoothening data.png')

# Apply Gaussian smoothing to reduce noise (optional but recommended)
blurred_image = cv2.GaussianBlur(image, (5, 5), 0)

# Create a Laplacian kernel for sharpening
laplacian_kernel = np.array([[0, -1, 0],
                             [-1, 5, -1],
                             [0, -1, 0]], dtype=np.float32)

# Apply the Laplacian filter for sharpening
sharpened_image = cv2.filter2D(blurred_image, -1, laplacian_kernel)

# Display the original image, blurred image, and sharpened image
cv2_imshow(image)
cv2_imshow(blurred_image)
cv2_imshow(sharpened_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

output: