**INDIVIDUAL REPORT 21R241-SHUDHI RISHAA P**

**Aim:**

A) Replicate the clock image to 03:25 using shapes approximately.Have the text as 03:25 PM

B) Create a white image of size 480\*480 and draw a Blue coloured polygon on it and Write “AI &amp; Vision” on the right corner and left corner of the given image.

# **Software/ Packages Used:**

1. Pycharm IDE
2. Libraries used:
   * NumPy
   * opencv-python
   * matplotlib
   * tkinter
   * Math

# **A)** **Replicate the clock image to 03:25 using shapes approximately.Have the text as 03:25 PM**

# **Introduction:** This report presents a Python code that utilizes the OpenCV library to create a simple analog clock image displaying the time as 03:25 PM. The code generates a white image with a black-outlined clock face, clock hands pointing to 3 and 5, and the corresponding time text.

# **Theory:**

# **1. Blank White Image:**

# A blank white image of size 300x300 pixels with three color channels (RGB) is created using NumPy. The intensity values are set to 255 to represent white.

# **2. Clock Center and Radius:**

# The center of the clock is determined based on the image size, and the radius is set to be slightly smaller than half of the minimum dimension.

# **3. Drawing Clock Circle:**

# The cv2.circle function is used to draw a black-outlined circle representing the clock face. The center, radius, color (black), and thickness (2) are specified.

# **4. Drawing Clock Hands:**

# The clock hands (hour and minute hands) are drawn using the cv2.line function.

# Hour Hand: The angle is calculated based on the current hour and a constant offset, and its position is determined using polar coordinates.

# Minute Hand: The angle is calculated based on the current minute, and its position is determined similarly.

# **5. Adding Time Text:**

# The time text "03:25 PM" is added using the cv2.putText function. The font, size, color (black), thickness, and position are specified.

# **6. Displaying the Clock Image:**

# The generated clock image is displayed using the cv2.imshow function, and the display window is named 'Clock Image'.

# **PROGRAM:**

# import cv2 import numpy as np import math # Create a blank white image img\_size = (300, 300, 3) clock\_image = np.ones(img\_size, dtype=np.uint8) \* 255 # Define clock center and radius center = (img\_size[1] // 2, img\_size[0] // 2) radius = min(center) - 10 # Draw clock circle cv2.circle(clock\_image, center, radius, (0, 0, 0), 2) # Draw clock hands hour\_angle = -90 + (3 % 12) \* 30 + 25 \* 0.5 minute\_angle = -90 + 25 \* 6 hour\_hand\_length = int(radius \* 0.5) minute\_hand\_length = int(radius \* 0.8) hour\_hand\_x = center[0] + int(hour\_hand\_length \* math.cos(math.radians(hour\_angle))) hour\_hand\_y = center[1] + int(hour\_hand\_length \* math.sin(math.radians(hour\_angle))) minute\_hand\_x = center[0] + int(minute\_hand\_length \* math.cos(math.radians(minute\_angle))) minute\_hand\_y = center[1] + int(minute\_hand\_length \* math.sin(math.radians(minute\_angle))) cv2.line(clock\_image, center, (hour\_hand\_x, hour\_hand\_y), (0, 0, 0), 4) cv2.line(clock\_image, center, (minute\_hand\_x, minute\_hand\_y), (0, 0, 0), 2) # Add text for the time font = cv2.FONT\_HERSHEY\_SIMPLEX cv2.putText(clock\_image, '03:25 PM', (50, 280), font, 1, (0, 0, 0), 2, cv2.LINE\_AA) # Display the clock image cv2.imshow('Clock Image', clock\_image) cv2.waitKey(0) cv2.destroyAllWindows()

# **OUTPUT:**

# The resulting image visually represents a simple analog clock with hands pointing to 3 and 5 and the time displayed as "03:25 PM".

# 

# **CONCLUSION:**

# The provided code demonstrates the use of OpenCV and NumPy to create a basic analog clock image. Understanding the principles of drawing shapes, calculating positions based on angles, and adding text contributes to fundamental knowledge in computer vision and image processing. Further customization can be applied to enhance the clock's appearance or make it interactive. This code serves as a starting point for creating more sophisticated clock applications or integrating clock elements into other projects.

**B) Create a white image of size 480\*480 and draw a Blue coloured polygon on it and Write “AI &amp; Vision” on the right corner and left corner of the given image.**

**INTRODUCTION:**

This report discusses a Python code that utilizes the OpenCV library to create a white image of size 480x480, draw a blue-colored polygon on it, and add the text "AI & Vision" in both the right and left corners of the image.

**THEORY:**

**1. Creating a White Image:**

* The NumPy library is employed to generate a blank white image with a size of 480x480 pixels and three color channels (RGB).
* The intensity values are set to 255 to represent white in each channel**.**

**2. Drawing a Blue-Colored Polygon:**

* The code uses NumPy to define vertices (pts) of a polygon: [(100, 100), (200, 300), (400, 200)].
* The vertices are reshaped to a NumPy array and used to fill the polygon with a blue color (255, 0, 0) using the cv2.fillPoly function.

**3. Adding Text - "AI & Vision":**

* The cv2.putText function is employed to write the text "AI & Vision" on the image.
* For the right corner: The position (340, 30) is specified with a font size of 0.8, black color (0, 0, 0), thickness of 2, and using the cv2.LINE\_AA anti-aliasing flag.
* For the left corner: The position (30, 30) is specified similarly.

**4. Displaying the Image:**

* The final image is displayed using cv2.imshow.
* The window title is set as 'Image with Polygon and Text'.
* The cv2.waitKey(0) command waits for a keyboard event before closing the window.
* The cv2.destroyAllWindows() command closes all OpenCV windows.

# **PROGRAM:**

# import cv2 import numpy as np # Create a white image of size 480x480 image\_size = (480, 480, 3) white\_image = np.ones(image\_size, dtype=np.uint8) \* 255 # Draw a blue-colored polygon on the image pts = np.array([[100, 100], [200, 300], [400, 200]], np.int32) pts = pts.reshape((-1, 1, 2)) cv2.fillPoly(white\_image, [pts], (255, 0, 0)) # Blue color # Write "AI & Vision" on the right corner font = cv2.FONT\_HERSHEY\_SIMPLEX cv2.putText(white\_image, 'AI & Vision', (340, 30), font, 0.8, (0, 0, 0), 2, cv2.LINE\_AA) # Write "AI & Vision" on the left corner cv2.putText(white\_image, 'AI & Vision', (30, 30), font, 0.8, (0, 0, 0), 2, cv2.LINE\_AA) # Display the image cv2.imshow('Image with Polygon and Text', white\_image) cv2.waitKey(0) cv2.destroyAllWindows()

# **OUTPUT:**

# 

# **Conclusion:**

# This code demonstrates the capabilities of OpenCV for creating and manipulating images. It showcases the drawing of polygons, adding text, and combining these elements to create visually appealing images. Understanding these fundamental concepts is essential for more complex image processing tasks and computer vision applications. The provided code serves as a starting point for creating customized images with specific shapes and text in the OpenCV environment.

# 

# **RESULT:**

a python program for convolution operation for the given pixel and kernel values were written and verified.