**CONVERSION OF IMAGE TO PEN-SKETCH**

import cv2

image = cv2.imread("ganesha.jpg")

grey\_filter = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

invert = cv2.bitwise\_not(grey\_filter)

blur = cv2.GaussianBlur(invert, (21,21),0)

invertedblur = cv2.bitwise\_not(blur)

sketch\_filter = cv2.divide(grey\_filter, invertedblur, scale=256.0)

cv2.imwrite("output.png",sketch\_filter)

cv2.imwrite("grey.png",grey\_filter)

cv2.imwrite("invert.png",invert)

cv2.imwrite("blur.png",blur)

cv2.imwrite("invertedblur.png",invertedblur)

(Open Source Computer Vision) is a popular computer vision library that provides tools for image and video processing

**Original pic**

OpenCV's **imread** function to read an image file named "ganesha.jpg" into a variable named **image**. This function reads an image from a file and returns it as a NumPy array.

**Grey pic**

OpenCV's **cvtColor** function to convert the image from the BGR color space to grayscale.

**Invert**

OpenCV's **bitwise\_not** function to invert the pixels of a grayscale image. The **bitwise\_not** operation inverts each bit of the input image, effectively swapping black and white pixels.

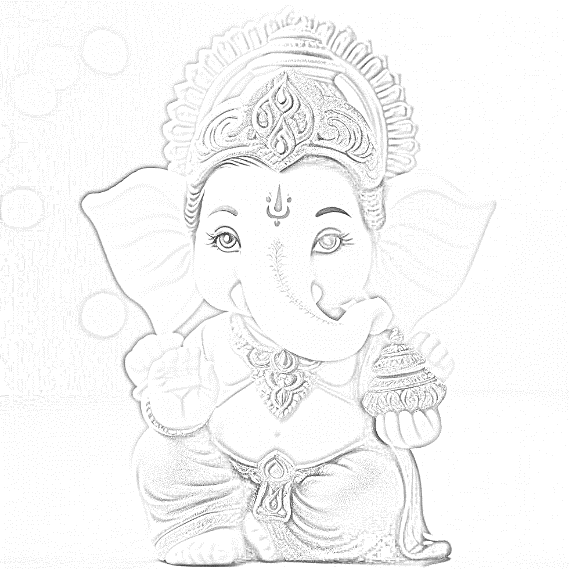
**Blur**

The Gaussian blur is a common image processing operation that smoothens an image by averaging the pixel values in a local neighborhood.

* **invert**: The inverted image that you obtained previously.
* **(21, 21)**: The size of the kernel used for blurring. In this case, it's a 21x21 kernel, which determines the extent of the neighborhood used for averaging pixel values.
* **0**: The standard deviation of the Gaussian kernel. In this case, it's set to 0, meaning that the standard deviation is computed based on the kernel size

**Inverted Blur**

the variable **invertedblur** contains the inverted version of the image after applying the Gaussian blur.

**Output**

sketch-like effect by dividing the grayscale image (**grey\_filter**) by the inverted and blurred image (**invertedblur**) using OpenCV's **divide** function. The **scale** parameter is set to **256.0** to normalize the result.

the variable **sketch\_filter** contains the result of the division operation, and it's saved as an image file named "output.png."