**CV Practical No.: 1**

**Aim: Introduction to OpenCV**

**New concept:**

**i. cv2:** **it** is the name of the module that belongs to OpenCV (Open Source Computer Vision Library). OpenCV is a popular library used for image and video processing, as well as machine learning applications related to computer vision. The cv2 module provides a range of functions for handling images, videos, and various computer vision tasks.

**ii. imread:** it is used to read an image from a specified file path. It loads the image into a format that can be processed by OpenCV, typically as a NumPy array.

**iii. cvtColor:** it is used to convert an image from one color space to another. It is used for converting between formats like RGB (Red, Green, Blue) and grayscale, or from BGR (Blue, Green, Red) to RGB, as OpenCV uses BGR by default.

**iv: imshow:** it is used to display an image in a window. It is used to show the result of some image processing operations or to visualize an image you’ve loaded.

**v. subplot:** is a function from the matplotlib library (not OpenCV) that allows you to create a grid of subplots within a single figure. We can display multiple images or plots in the same window using different subplots.

**Theory:**

**Computer Vision** is a field of artificial intelligence (AI) that enables computers to interpret and understand the visual world. It involves techniques for processing and analyzing images, videos, and visual data to make decisions or generate insights. Computer vision tasks often rely on machine learning and deep learning methods to automate the interpretation of visual data.

Applications of Computer Vision (CV):

1. **Healthcare & Medical Imaging:** Used to detect diseases like cancer, analyze X-rays and MRIs, and assist in surgeries.
2. **Autonomous Vehicles:** Helps self-driving cars detect objects, recognize lanes, and read traffic signs.
3. **Retail & E-Commerce:** CV helps with inventory tracking, product recognition, and checkout-free shopping (e.g., Amazon Go).
4. **Agriculture:** Used for crop monitoring, automated harvesting, and weed detection.
5. **Surveillance & Security:** Includes facial recognition, anomaly detection, and automated security monitoring.

**Program:**

**Program 1: To show the image:**

import cv2

img = cv2.imread("bheem.jpg")

cv2.imshow("Output Image", img)

**Program 2: To convert rgb image to grayscale image:**

**Method 1:**

import cv2

img = cv2.imread("bheem.jpg")

gray\_image = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

cv2.imshow("Original Image", img)

cv2.imshow("Grayscale Image", gray\_image)

**Method 2:**

import cv2

img = cv2.imread("bheem.jpg")

gray\_image = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

cv2.imshow("Original Image", img)

converted\_image = cv2.imread("bheem.jpg", cv2.IMREAD\_GRAYSCALE)

cv2.imshow("Converted Image", converted\_image)

**Program 3: To display multiple images on same screen:**

import cv2

import matplotlib.pyplot as plt

img = cv2.imread("bheem.jpg")

gray\_image = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

#cv2.imshow("Original Image", img)

#cv2.imshow("GrayScale Image", gray\_image)

#Image1

plt.figure(figsize = (10,5))

plt.subplot(1,2,2)

plt.imshow(img)

#plt.imshow(img, cmap = 'gray')

plt.title('Original Image')

plt.axis('off')

#Image2

plt.subplot(1,2,1)

plt.imshow(gray\_image)

#plt.imshow(gray\_image, cmap = 'gray')

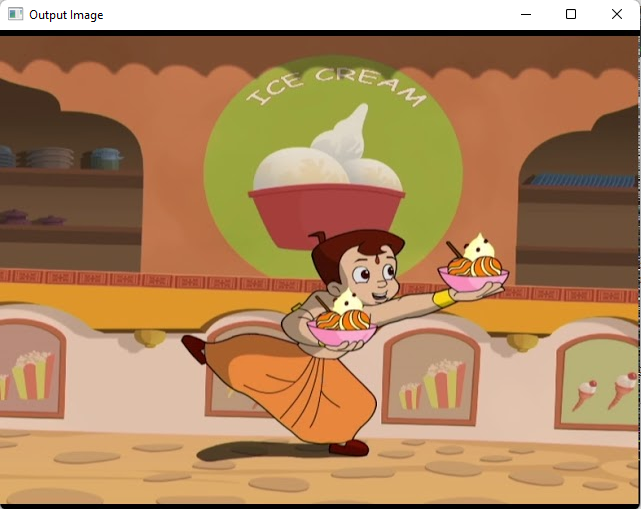
plt.title('GrayScale Image')

plt.axis('off')

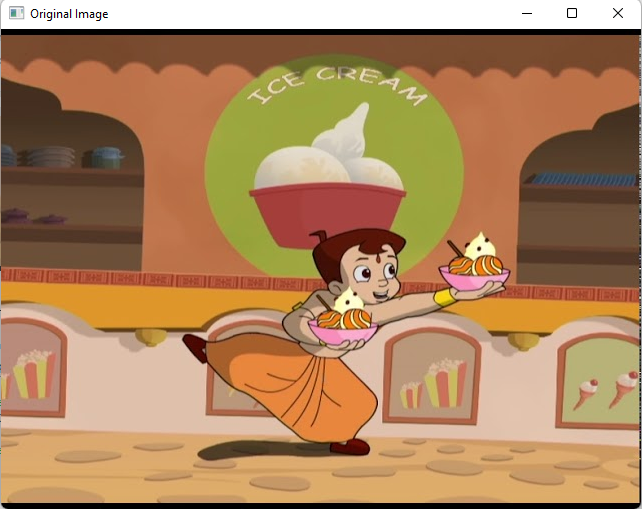
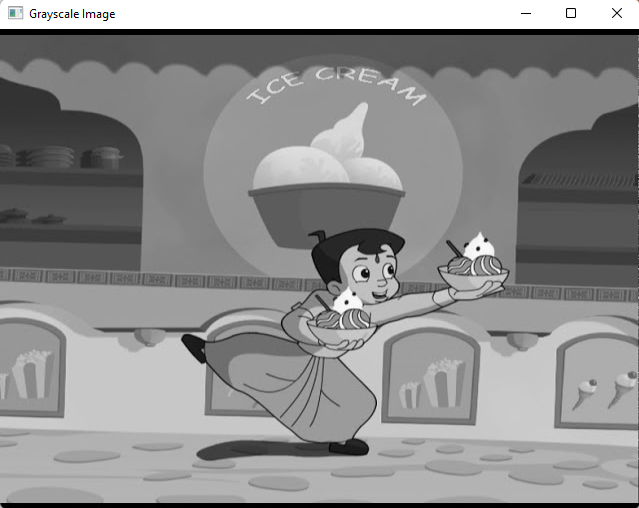
plt.show()

**Output:**

**Program 1:**



**Program 2:**

**Program 3:**

