**Image Processing Task**

**Assignment no 1**

**Introduction**

The code you've provided performs several image processing tasks using the OpenCV library in Python. Each task demonstrates a different aspect of image manipulation and processing. The code works with a sample image ('bird.jfif') and 'Coins.PNG') to showcase various image processing techniques. Let's explore each task in detail.

**Task 1: Changing Resolution**

In this task, the code reads an image ('bird.jfif') and resizes it to a resolution of 256x256 pixels. This task demonstrates how to change the size of an image using the `cv.resize()` function. The resized image is displayed using `cv.imshow()`.

**Task 2: Grayscale Conversion**

Task 2 involves converting the resized image from Task 1 into grayscale. Grayscale images contain only shades of gray, making them suitable for various image processing tasks. The code uses `cv.cvtColor()` to perform the RGB to grayscale conversion and then displays the grayscale image.

**Task 3: RGB to Binary Conversion**

This task focuses on converting the resized RGB image from Task 1 into a binary image. Binary images consist of only two values, typically black and white, and are useful for image segmentation. The code sets specific RGB color range bounds (lower and upper bounds) for white color and uses `cv.inRange()` to create a binary mask. The original RGB image and the binary image are displayed side by side using `cv.imshow()`.

Additionally, the code performs grayscale to binary conversion by applying a binary threshold using `cv.threshold()` on the grayscale image created in Task 2.

**Task 4: Contours and Object Identification**

In Task 4, the code works with a different image ('Coins.PNG') to demonstrate contour detection and object identification. The image is preprocessed to reduce noise using Gaussian blur (`cv.GaussianBlur()`) to remove small gaps between objects. Then, it's converted to grayscale.

The code uses contour detection (`cv.findContours()`) to find boundaries of objects in the preprocessed image. Contours represent the outlines of objects in an image. These contours are drawn onto a copy of the original image using `cv.drawContours()`.

To identify specific objects (coins in this case), a minimum and maximum area filter is applied to the detected contours. Objects with areas falling within the specified range are considered as coins. The number of coins is counted, and the result is printed.

**Conclusion**

In conclusion, this code demonstrates a series of image processing tasks, including resizing, color space conversion, binary image creation, and object identification using contours. These tasks showcase fundamental image processing techniques used in computer vision and image analysis. By combining these techniques, you can perform a wide range of tasks, from basic image manipulation to more complex object recognition and segmentation.