**Assignment 02 Report**

## Question 1: Determining the Time Shown by an Analog Clock

For this question, we were tasked with developing image processing techniques to determine the current time shown by an analog clock. The suggested approach involved segmenting the hour and minute hands and calculating the angles between them.

To achieve this, we used the following steps:

1. **Image Preprocessing:** The input image of the analog clock was preprocessed to enhance the contrast and reduce noise. This step involved resizing, applying filters (e.g., Gaussian blur), and thresholding.
2. **Edge Detection:** We performed edge detection using techniques such as Canny edge detection to identify the clock's hands' edges.
3. **Line Detection:** We applied the Hough Line Transform to detect lines representing the clock's hands. The two most prominent lines represented the hour and minute hands.
4. **Angle Calculation:** We calculated the angles between these lines to determine the positions of the hour and minute hands, thereby estimating the current time.

The result of our image processing techniques is an accurate estimation of the time shown on the clock. These techniques can be used for various applications where time reading from analog clocks is required.

## Question 2: Character Matching and Bounding Box

In this question, we were tasked with creating a code that can identify a matching character in the lower section when different characters are displayed one at a time in the upper section of the image. When a match is found, a green bounding box is drawn around the corresponding matched character box(es). If no match is discovered, the code should display the message "No match found."

The approach involved:

1. **Template Matching:** We used template matching techniques to compare the character in the upper section with templates stored in the lower section. When a match was found, the coordinates of the match were recorded.
2. **Drawing Bounding Boxes:** For each match found, we drew a green bounding box around the matched character's location.
3. **No Match Detection:** If no match was found after processing all characters, the code displayed the message "No match found."

This code can be applied in various scenarios, such as optical character recognition (OCR) or text detection in images.

## Question 3: Score Calculation from Image

In this question, we were tasked with creating a program to determine the score of a student in reading, writing, and speaking tests using bars displayed in an image. The program was required to display the scores on the image itself.

The steps to achieve this were as follows:

1. **Image Preprocessing:** The input image with score bars was preprocessed to enhance the visibility of the bars and text using techniques like resizing, thresholding, and contour detection.
2. **Bar Detection:** We detected the individual bars and extracted the information from the text displayed at the end of each bar.
3. **Score Calculation:** We calculated the scores based on the information extracted from the bars and displayed the scores on the image.

This program provides an automated way to extract and calculate scores from images, which can be useful in educational and evaluation settings.

## Question 4: Comprehensive Image Processing Software

For this question, we were required to create a comprehensive image processing software with a user-friendly interface and a wide range of functionalities. The software includes the following features:

* **Image Loading and Saving:** The software enables users to load images and save processed images.
* **Brightness Adjustment:** Users can adjust the brightness of images.
* **Log Operations:** The software allows various log operations with customizable parameter values.
* **Thresholding:** Thresholding is supported for both color and grayscale images.
* **Color Conversion:** The software can convert images between different color spaces.
* **Drawing Shapes:** Users can draw shapes on the image using a mouse interface.
* **Blurring:** The software provides various blurring filters for image smoothing.
* **Image Sharpening:** It implements Laplacian-based image sharpening techniques.
* **Unsharp Masking:** Unsharp masking for enhancing image details is supported.
* **Color-Based Image Segmentation:** The software enables segmentation based on color.
* **Line Detection:** It uses the Hough Transform to detect lines in images.
* **Morphological Operations:** Erode, Dilate, Open, and Close operations are available.
* **Connected Component Analysis:** The software can perform analysis of connected components in an image.
* **Contour Detection:** Users can detect contours in images, including area, smallest object, largest object, and hole detection. The software displays object details upon clicking with the mouse.
* **Time Determination:** The function developed for Question 1 can be called to determine the time from an analog clock image.
* **Character Matching:** The function developed for Question 2 can be called for character matching.
* **Score Calculation:** The function developed for Question 3 can be used for score calculation from bar graphs.

In addition, to ensure ease of execution on any system, the software is packaged using a Docker container.

This comprehensive software serves as a powerful tool for image processing tasks, offering a wide range of features for various applications, from basic image editing to more advanced image analysis.

**Conclusion**

In this assignment, we have successfully developed image processing techniques and a comprehensive image processing software with a broad range of features. These solutions can be applied to various real-world scenarios, from analyzing analog clocks to character matching and image manipulation. The use of a Docker container ensures the software's portability and ease of execution across different systems.