# Image segmentation using K-means clustering algorithm

## 1. Importing necessary libraries

First, we need to import the libraries required for our analysis: `numpy` for numerical operations, `matplotlib` for plotting images, `cv2` (OpenCV) for image processing, and `sklearn` for implementing the K-means algorithm for image segmentation.

```
import numpy as np
import matplotlib.pyplot as plt
import cv2
from sklearn.cluster import KMeans
```

## 2. Uploading the image

Next, we upload an image file from our local machine to the Google Colab environment.

```
from google.colab import files
uploaded = files.upload()
```

Choose Files sample1.jpg

- **sample1.jpg**(image/jpeg) - 7571 bytes, last modified: 2/22/2025 - 100% done

```
Saving sample1.jpg to sample1 (1).jpg
```

## 3. Preprocessing the image

The uploaded image is then read using OpenCV and converted from BGR to RGB format for proper visualisation.

```
image_path = next(iter(uploaded.keys()))
image = cv2.imread(image_path)
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Displaying the original image
plt.figure(figsize=(8,6))
plt.imshow(image_rgb)
plt.axis('off')
plt.title('Original Image')
plt.show()
```

Original Image

## 4. Reshaping the image

Now, the image is reshaped into a 2D array where each row corresponds to a pixel's RGB values. This format is necessary for applying clustering algorithms, as they operate on 2D arrays.

```
pixel_values = image_rgb.reshape((-1, 3))
pixel_values = np.float32(pixel_values)
```

## 5. Implementing the K-means clustering algorithm

We apply K-means clustering to segment the image. We define the number of clusters, and the `KMeans` class from `sklearn.cluster` performs the clustering. The resulting labels are reshaped back to the original image dimensions to visualise the segmented image.

```
num_clusters = 5

kmeans = KMeans(n_clusters=num_clusters, random_state=42)
kmeans_labels = kmeans.fit_predict(pixel_values)

kmeans_segmented_image = kmeans_labels.reshape(image_rgb.shape[:2])
```

## 6. Visualising the result

Finally, we plot the results of the segmentation.

```
plt.figure(figsize=(12,6))

plt.imshow(kmeans_segmented_image, cmap='viridis')
plt.axis('off')
plt.title('K-means Segmentation')
plt.show()
```