Vector Quantization

(Using LBG Algorithm with Splitting)

# **Summary:**

We start by **dividing** the image **into blocks (vectors)**, then we generate **Best “K” Vectors** that can be used to **re-Construct** the original Image, these vectors are called **Codebook**. For each Block in the Image, Select the **Nearest** vector (using Euclidean Distance), then **label** each Block in the image with **INDEX** of Nearest Vector (in the Codebook).

In order to **re-Construct** the Image, it is required to have:   
● **All Labels** (one label for each BLOCK in the Image).  
● **The Codebook itself** which consists of K Vectors, each vector is a small Image with size equal to BLOCK size.

The project is built using Java, **Intellij**.

# **Example:**

* The original **GRAY** image is **600\*600 pixels** (each pixel is saved in one byte)
* The image is divided into **Blocks each of size 4\*4 pixels**
* **The Codebook** (which will be used to Reconstruct the image) consists of 32 Vectors **(32 blocks each of size 4\*4)**

**Solution:**

* **Number Blocks in the image** = (600\*600)/ (4\*4) =22500 Blocks
* **Number of labels** = Number of Blocks = 22500 Labels
* As **Number of Vectors in the codebook = 32**, Indexes will range from 0 to 31 (from 00000 to 11111 Binary) In other words, **each index** can be **saved in 5 Bits** (each label is 5 bits)

**Compression Ratio:**

* **Labels Size** = Number of Labels \* bits/Label = 22500 \* 5 bits = 112500 bits (14063 Bytes)
* **Codebook Storage Size** = Number of Vectors \* Vector Size (in pixels) \* number of bits to save a **pixel** = 32 \* (4\*4) \* 8 bits = 4096 bits (512 bytes)
* **Total Compressed Image Size** = Label Size + Code book storage size = 112500+ 4096 = 116596 bits (14575 bytes)
* **Original Image Size** = 600 \* 600 (pixels) \* 8 bits/pixel = 2880000 bits (360000 Bytes)
* **Compression Ratio** = Original / Compressed = 360000/14575 = **24.7:1**







