**HO CHI MINH CITY NATIONAL UNIVERSITY**

UNIVERSITY OF NATURE SCIENCE

FACULTY OF INFORMARTION TECHNOLOGY

---------------o0o---------------

**Logo

Description automatically generated**

**PROJECT 1:**

**COLOR COMPRESSION**

**Lecturers:**

Vũ Quốc Hoàng

Phan Thị Phương Uyên

Nguyễn Văn Quang Huy

**Students:**

Đặng Ngọc Tiến ID: 20127641

Contents

[I. Introduction 1](#_Toc106093600)

[1. Personal information: 1](#_Toc106093601)

[II. Implementation idea and description 1](#_Toc106093602)

[1. Implementation idea 1](#_Toc106093603)

[2. Description 1](#_Toc106093604)

[III. Demo and Comment results 3](#_Toc106093605)

[IV. References 4](#_Toc106093606)

# Introduction

## Personal information:

Name: Đặng Ngọc Tiến

ID student: 20127641

# Implementation idea and description

## Implementation idea

**A recipe for k-means**

* Decide how many clusters you want, i.e., choose k
* Randomly assign a centroid to each of the k clusters
* Calculate the distance of all observation to each of the k centroids
* Assign observations to the closest centroid
* Find the new location of the centroid by taking the mean of all the observations in each cluster
* Repeat steps 3-5 until the centroids do not change position

## Description

* Use the PIL library to read the image, then reshape the image from a three-dimensional scalar into a 2-dimensional array of the form [height \* width, n], where = 3

Text

Description automatically generated

* For each parameter passed in init centroids is random in the range [0.255] or random pick from the color of the image, we perform initialization for centroids as k color points with k being the number of clusters.

A screenshot of a computer

Description automatically generated

* Calculate the distance using np.linalg.norm in Numpy library



* Assign observations to the closest centroid using np.argmin in Numpy library

A screenshot of a computer

Description automatically generated with medium confidence

* Update centroid location.

Text

Description automatically generated

* And repeat by max\_iter
* Return centroids and labels

**Test program**

* After returning labels and centroids in kmean then we assign them back to the image

Text

Description automatically generated

* And using matplotlib library to display img

# Demo and Comment results

* **Test program:**

Graphical user interface, website

Description automatically generated

* **Comments:**
  + In general, the demo results give quite good results, if compared with sciket-learn's Kmean, we have approximately the same efficiency. However, the image compression performance for small images (compressed - reduced color by other software) or monochrome images often produces bad results with low max\_iter.
  + In addition, if we plot the convergence of Kmean, when we init 'random' the convergence speed will be faster than the init taken from the point on the image in some cases. In case the image has many colors, taking random gives better performance.

# References

* Idea Kmean algorithm:

[K-means Clustering in Python: A Step-by-Step Guide (dominodatalab.com)](https://blog.dominodatalab.com/getting-started-with-k-means-clustering-in-python)

* Library:

[NumPy Documentation](https://numpy.org/doc/)

[Matplotlib documentation — Matplotlib 3.5.2 documentation](https://matplotlib.org/stable/index.html)

[Pillow — Pillow (PIL Fork) 9.1.1 documentation](https://pillow.readthedocs.io/en/stable/)