

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**  
**ĐẠI HỌC KHOA HỌC TỰ NHIÊN**  
**NGÀNH CÔNG NGHỆ THÔNG TIN**



# **PROJECT 01**

Hoàng Nguyên Trúc

18127055

**Môn học: Toán ứng dụng và thống kê**  
**Thành phố Hồ Chí Minh – 2020**

---

# MỤC LỤC

---

MỤC LỤC.....	2
Ý TƯỞNG THỰC HIỆN.....	3
HÌNH ẢNH VỚI TỪNG SỐ LƯỢNG MÀU.....	6
NHẬN XÉT .....	8
TÀI LIỆU THAM KHẢO.....	9

## Ý TƯỞNG THỰC HIỆN

### 1) Hàm chuyển đổi hình ảnh → ma trận :

```
def data(path):  
    img = Image.open(path).convert('RGB')  
    img_npa = np.asarray(img) # convert to numpy array  
  
    shape = np.array(img_npa).shape  
    width = shape[0]  
    height = shape[1]  
    number_chanel = shape[2] #number of main colors RGB  
  
    img_1d = np.reshape(img, (width*height, number_chanel))  
    return width, height, img_1d
```

- Input: nhận đường dẫn tới hình ảnh.
- Đọc ảnh và chuyển mode từ 'RGBA' sang 'RGB'.
- Chuyển đổi hình ảnh sang numpy array.
- Chuyển ma trận 3 chiều → 2 chiều.
- Lấy chiều dài, chiều rộng của ảnh.

## 2) Những hàm tính toán vector:

```
def add_vector(v, w):  
    return [vi + wi for vi, wi in zip(v, w)]  
  
def sub_vector(v, w):  
    v = np.array(v, dtype = 'int')  
    w = np.array(w, dtype = 'int')  
    return [vi - wi for vi, wi in zip(v, w)]  
  
def inner_product(v, w):  
    return sum(vi*wi for vi, wi in zip(v, w))  
  
def norm_square(v):  
    return inner_product(v, v)  
  
def norm(v):  
    return math.sqrt(norm_square(v))
```


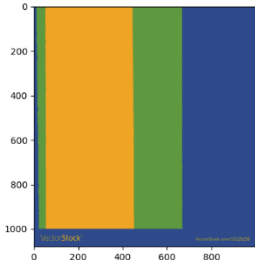
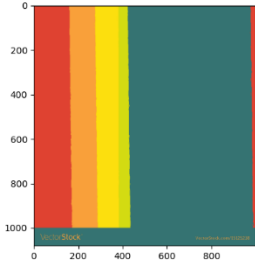
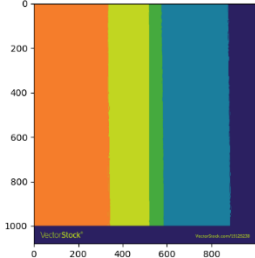
Dựa vào những hàm trong Notebook Lab01-Moodle.


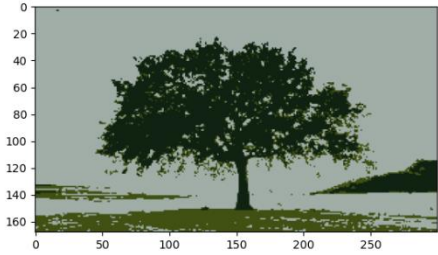
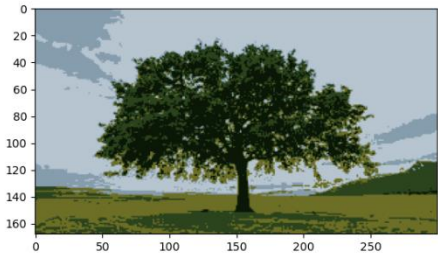
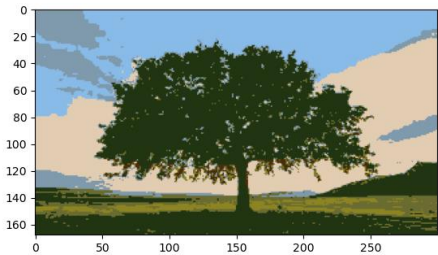
### 3) Hàm kmean:

```
def kmeans(img_id, k_clusters, max_iter, init_centroids='random'):  
    # Step 1: get centers  
    centers = []  
    if init_centroids == 'in_pixels':  
        centers = img_id[np.random.choice(img_id.shape[0], k_clusters, replace=False)]  
    elif init_centroids == 'random':  
        temp = []  
        while len(temp) < k_clusters:  
            ele = []  
            for _ in range(3):  
                num = random.randint(0,255)  
                ele.append(num)  
                if ele not in temp:  
                    temp.append(ele)  
            centers = np.array(temp, dtype = 'uint8')  
  
    # Step 2: Calculate distance from each pixel to k centers then pick min value (min value = a group that pixel belongs to), assign label for each pixels.  
    temp = img_id.copy()  
    labels = [1 for i in range(img_id.shape[0])]   
    while max_iter > 0:  
        for index,element in enumerate(img_id):  
            dis = [norm(sub_vector(i, img_id[index])) for i in centers]  
            labels[index] = np.argmin(dis)  
  
        old_center = [c for c in centers]  
        new_old_center = np.array(old_center)  
  
        #Step 3: Calculate new centers  
        for ki in range(k_clusters):  
            group = []  
            for index,element in enumerate(img_id):  
                if labels[index] == ki:  
                    group.append(element)  
            new_group = np.array(group)  
            avg = np.mean(new_group, axis = 0)  
            centers[ki] = avg  
  
        # Compare whether if new centers equal to old center or not. If yes, kmean function will stop and vice versa.  
        if (set([tuple(a) for a in new_old_center]) == set([tuple(a) for a in centers])): #check if old_center == new_center  
            break  
        max_iter-=1  
  
        # Assign new value for each pixel based on label value.  
        for l in range(len(labels)):  
            for k in range(k_clusters):  
                if labels[l] == k:  
                    temp[l] = centers[k]  
  
    return labels, centers, temp
```

- 1) Lấy centers theo 2 cách: in\_pixels hoặc ngẫu nhiên giá trị bất kì trong khoảng (0 → 255).
- 2) Tính khoảng cách từng phần tử tới center, lấy giá trị khoảng cách nhỏ nhất và thêm phần tử đó vào nhóm cluster gần nhất.
- 3) Cập nhật centers mới.
- 4) Chạy lại bước 1, bước 2 tới khi centers cũ và centers mới bằng nhau.

# HÌNH ẢNH VỚI TỪNG SỐ LƯỢNG MÀU

	
$K = 3 (\text{In\_pixels}):$	
$K = 5 (\text{In\_pixels}):$	
$K = 7 (\text{In\_pixels}):$	

	
$K = 3(\text{In\_pixels}):$	
$K = 5(\text{In\_pixels}):$	
$K=7(\text{In\_pixels}):$	

## NHẬN XÉT

- Với số lượng  $k$  càng lớn thì hình sẽ càng rõ hơn vì số lượng nhóm màu tăng.
- Với số lần lặp càng lớn thì độ chính xác sẽ cao hơn.
- In\_pixels sẽ cho ra màu chính xác hơn random vì có trường hợp random sẽ tạo ra màu không có trong màu của ảnh dẫn tới ảnh bị mất nhóm màu  $\rightarrow$  ảnh được tạo ra không rõ và ít màu.

	In_pixels	Random
		
$K = 3$		
$K = 5$		
$K = 7$		



---

## TÀI LIỆU THAM KHẢO

---

- 1) Notebook Lab01-Moodle.
- 2) Notebook Lab02-Moodle.
- 3) [https://machinelearningcoban.com/2017/01/01/kmeans/?fbclid=IwAR1BhXz6h\\_KL3afgA7eTKu3SSyFBq6YAVV2jG9DTvOgtBRRSlyOWo8tJgz8#-gioi-thieu](https://machinelearningcoban.com/2017/01/01/kmeans/?fbclid=IwAR1BhXz6h_KL3afgA7eTKu3SSyFBq6YAVV2jG9DTvOgtBRRSlyOWo8tJgz8#-gioi-thieu)