There is an image in the name of “tiger.png”. Use k means clustering with k set to 16 and cluster the image, which means that you want to keep just 16 colors in our compressed image.

Objective: Open and display the image “tiger.png”. Convert the image into numpy array, so that it can be used in further processing. Find out the dimensions of the image and convert it into a two dimensional array (U se k means clustering for image segmentation , reducing the image into 16 colors).

In [3]:
```
from sklearn.cluster import KMeans
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import os
%matplotlib inline
```

In [4]:
```
#Get image and its coresponding RGB values

img = Image.open("tiger.png")
img_np=np.asarray(img)
img_np[0:2]
```

Out[4]:
```
array([[[164, 160, 159],
        [165, 161, 160],
        [164, 163, 161],
        ...,
        [160, 128,  90],
        [158, 125,  90],
        [161, 128,  93]],

       [[164, 160, 159],
        [164, 160, 159],
        [163, 162, 160],
        ...,
        [164, 132,  94],
        [162, 129,  94],
        [157, 124,  89]]], dtype=uint8)
```

In [5]:
```
#Get image dimensions

img_np.shape
```

Out[5]:
```
(720, 1280, 3)
```

For feeding this data into the algorithm, we must change the shape of this data into a dataset with 720*1280=921600 rows and 3 columns

In [7]:
```
#Reshape data

pixels=img_np.reshape(img_np.shape[0]*img_np.shape[1],img_np.shape[2])
pixels.shape
```

Out[7]:
```
(921600, 3)
```

In [8]:
```
model=KMeans(n_clusters=16)
model.fit(pixels)
```

Out[8]:
```
KMeans(n_clusters=16)
```

In [9]:
```
#Define the cluster centers

pixel_centroids=model.labels_
cluster_centers=model.cluster_centers_
pixel_centroids
```

Out[9]:
```
array([15, 15, 15, ...,  8,  8,  8])
```

In [10]:
```
cluster_centers
```

Out[10]:
```
array([[107.61421125,  70.50572334,  44.41261689],
       [175.63423151, 176.29005412, 176.60338707],
       [ 50.83200039,  41.09377291,  35.55696904],
       [230.65614774, 229.57748259, 229.77485825],
       [ 61.1743609 , 126.52992481,  44.1073183 ],
       [129.62039864, 134.29246537, 121.88426633],
       [203.11299601, 200.7739591 , 199.4288764 ],
       [ 68.79798073,  67.09029959,  56.4914874 ],
       [107.04375366, 115.2010804 , 101.77238775],
       [ 24.40473298,  18.80548659,  16.55184618],
       [187.94512108,  85.12294025, 131.51196446],
       [135.70845119, 109.62655353,  83.83039056],
       [ 92.59143037,  89.73971802,  80.74263003],
       [211.97224441, 170.14869542, 135.02402822],
       [176.56846164, 137.69929531, 102.75160218],
       [152.80238141, 155.45275972, 153.59857891]])
```

In [12]:
```
#Cluster assigment

final=np.zeros((pixel_centroids.shape[0],3))
for cluster_no in range (16):
    final[pixel_centroids==cluster_no]=cluster_centers[cluster_no]
final[0:5]
```

Out[12]:
```
array([[152.80238141, 155.45275972, 153.59857891],
       [152.80238141, 155.45275972, 153.59857891],
       [152.80238141, 155.45275972, 153.59857891],
       [152.80238141, 155.45275972, 153.59857891],
       [152.80238141, 155.45275972, 153.59857891]])
```

In [14]:
```
#Reshape to riginal Dimensions

comp_image=final.reshape(img_np.shape[0],img_np.shape[1],3)
comp_image.shape
```

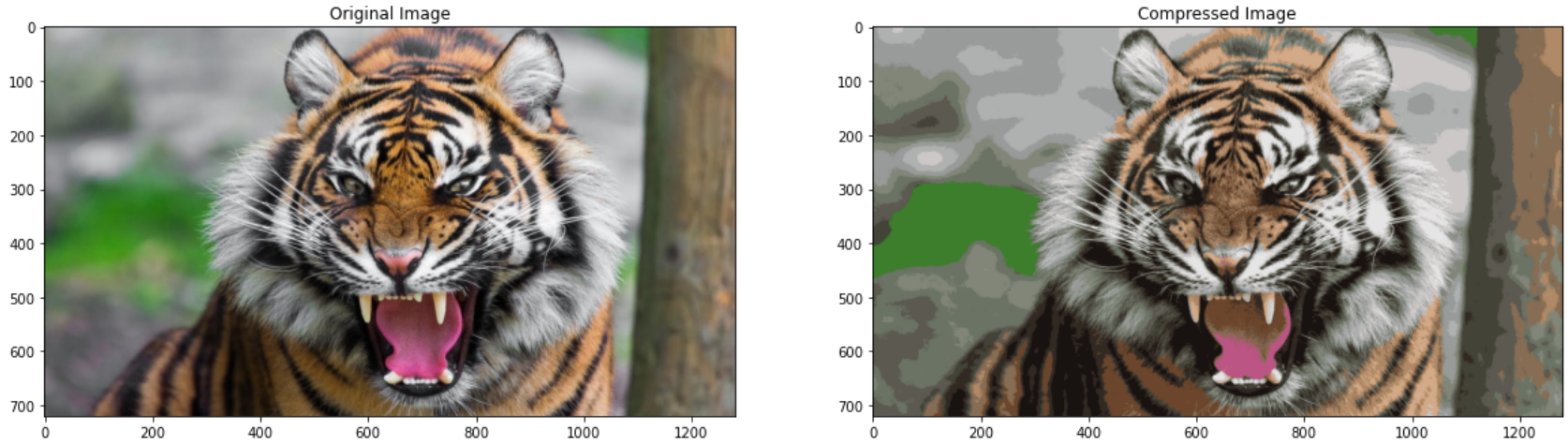
Out[14]:
```
(720, 1280, 3)
```

In [17]:
```
#convert the pixel values to image

comp_image=Image.fromarray(np.uint8(comp_image))
comp_image.save('tiger_compressed.png')
img_1=mpimg.imread('tiger.png')
img_2=mpimg.imread('tiger_compressed.png')
```

In [19]:
```
# origianl vs compressed Image

fig, (ax1,ax2)= plt.subplots(1,2, figsize=(20,20))
ax1.imshow(img_1)
ax1.set_title('Original Image')
ax2.imshow(img_2)
ax2.set_title('Compressed Image')
plt.show()
```



In [ ]: