# پروژه پایانی درس یادگیری ماشین

## استاد محترم جناب دکتر آهنگران - تهیه کننده : امین زایراومالی

***طبق تعریف پروژه سه تصویر نمونه داده شده بایستی توسط الگوریتم خوشه بندی کی مین برای تصاویر قطعه بندی شده رنگی به ازای مقادیر مختلف کا استفاده و اعمال گردد***

الگوریتم : k-means

مقادیر کا : k=2 , k=3 , k=5

***از مهمترین تکنیک های عملی داده کاوی که کاربرد زیادی در علوم مختلف دارد، می توان به خوشه بندی کی مین اشاره کرد خوشه‌بندی کی مین یک الگوریتم یادگیری ماشینی بدون نظارت است که هدف آن تقسیم تعداد مشخص مشاهدات به خوشه‌های کا است که در آن هر مشاهده متعلق به خوشه‌ای با نزدیک‌ترین میانگین است. خوشه به مجموعه ای از نقاط داده اشاره دارد که به دلیل شباهت های خاص با هم جمع شده اند. برای تقسیم‌بندی تصویر، خوشه‌ها در اینجا رنگ‌های تصویر متفاوتی دارند.***

پیاده سازی پروژه با زبان پایتون انجام شد به شرح کدهای زیر می باشد

## مرحله اول : اضافه کردن کتابخانه های مورد نیاز

In [1]:
```
import cv2 as cv                    # کتابخانه opencv
import numpy as np                  # کتابخانه numpy
import matplotlib.pyplot as plt     # کتابخانه matplotlib
import matplotlib.image as mpimg  # کتابخانه Ipython  تابع نمایش تصویر
```

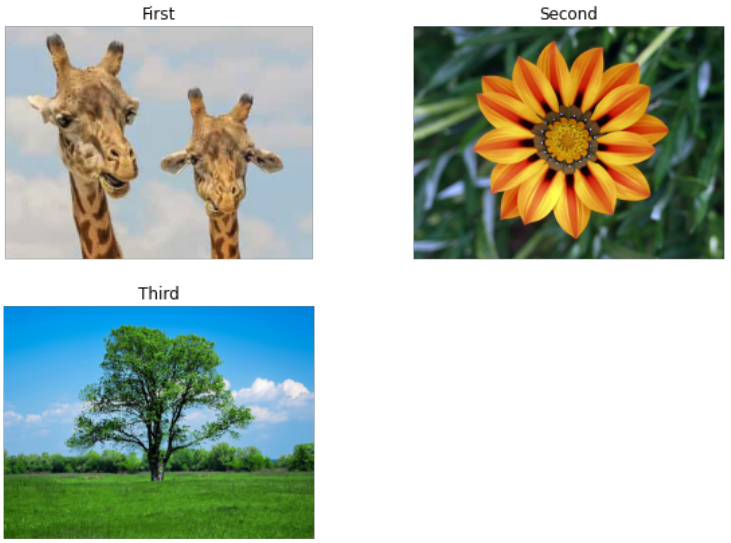
## مرحله دوم : لود کردن اطلاعات تصاویر در متغیر های مربوط به خودشان

In [2]:
```
# خواندن سه تصویر و قرار دادن در متغیر مربوطه اش
Orginal_img1 = mpimg.imread('image-1.jpg')
Orginal_img2 = mpimg.imread('image-2.jpg')
Orginal_img3 = mpimg.imread('image-3.jpg')
```

## مرحله سوم : نمایش تصاویر اصلی

In [3]:
```
# نمایش ۳ تصویر داده شده
fig = plt.figure(figsize=(10, 7))
rows = 2
columns = 2
fig.add_subplot(rows, columns, 1)
plt.imshow(Orginal_img1)
plt.axis('off')
plt.title("First")
fig.add_subplot(rows, columns, 2)
plt.imshow(Orginal_img2)
plt.axis('off')
plt.title("Second")
fig.add_subplot(rows, columns, 3)
plt.imshow(Orginal_img3)
plt.axis('off')
plt.title("Third")
```

Out[3]:
```
Text(0.5, 1.0, 'Third')
```




## مرحله چهارم : تبدیل تصاویر به صورت دو بعدی در فرمت سه رنگ آر جی بی RGB - 2 Dimention

In [4]:
```
#  RGB - 2D تبدیل تصاویر به دو بعدی و ۳ رنگ اصلی آر جی بی
image1 = Orginal_img1.reshape((-1,3))
image2 = Orginal_img2.reshape((-1,3))
image3 = Orginal_img3.reshape((-1,3))
```

مرحله ششم : تبدیل اطلاعات تصاویر به نوع فلوت Float

In [5]:
```
image1 = np.float32(image1)
image2 = np.float32(image2)
image3 = np.float32(image3)
```

مرحله هفتم : نمایش میزان پیکسل ردیف اطلاعات تصاویر

In [6]:
```
# نمایش میزان پیکس ردیف اطلاعات تصاویر
print("Image 1 : ", image1.shape)
print("Image 2 : ", image2.shape)
print("Image 3 : ", image3.shape)
```

```
Image 1 :  (77440, 3)
Image 2 :  (150528, 3)
Image 3 :  (231852, 3)
```

مرحله هشتم : تعیین تعداد خوشه بندی تصاویر برای سه میزان کا به تعداد ۲ و ۳ و ۵

In [7]:
```
k1 = 2
k2 = 3
k3 = 5
```

مرحله نهم : انجام خوشه بندی با الگوریتم برای تعداد ۲ دسته

In [8]:
```
# این معیار خاتمه تکرار است. وقتی این معیار برآورده شد، تکرار الگوریتم متوقف می شود
# علت استفاده برای این است که در حلقه مینیمال بی پایان قرار نگیرد
criteria = (cv.TERM_CRITERIA_EPS + cv.TERM_CRITERIA_MAX_ITER, 10, 1.0)

# خوشه بندی تصویر اول با سه مقدار کا تعیین شده
ret1_k1,label1_k1,center1_k1=cv.kmeans(image1,k1,None,criteria,10,cv.KMEANS_RANDOM_CENTERS)
ret1_k2,label1_k2,center1_k2=cv.kmeans(image1,k2,None,criteria,10,cv.KMEANS_RANDOM_CENTERS)
ret1_k3,label1_k3,center1_k3=cv.kmeans(image1,k3,None,criteria,10,cv.KMEANS_RANDOM_CENTERS)

# خوشه بندی تصویر دوم با سه مقدار کا تعیین شده
ret2_k1,label2_k1,center2_k1=cv.kmeans(image2,k1,None,criteria,10,cv.KMEANS_RANDOM_CENTERS)
ret2_k2,label2_k2,center2_k2=cv.kmeans(image2,k2,None,criteria,10,cv.KMEANS_RANDOM_CENTERS)
ret2_k3,label2_k3,center2_k3=cv.kmeans(image2,k3,None,criteria,10,cv.KMEANS_RANDOM_CENTERS)

# خوشه بندی تصویر سوم با سه مقدار کا تعیین شده
ret3_k1,label3_k1,center3_k1=cv.kmeans(image3,k1,None,criteria,10,cv.KMEANS_RANDOM_CENTERS)
ret3_k2,label3_k2,center3_k2=cv.kmeans(image3,k2,None,criteria,10,cv.KMEANS_RANDOM_CENTERS)
ret3_k3,label3_k3,center3_k3=cv.kmeans(image3,k3,None,criteria,10,cv.KMEANS_RANDOM_CENTERS)
```

مرحله دهم : تبدیل داده های خوشه بندی به تصاویر

In [9]:
```
# Now convert back into uint8, and make original image   ایجاد عکسهای اورجینال از داده های خوشه بندی شده

# تبدیل تصویر اول به ازای سه مقدار کا
center1_k1 = np.uint8(center1_k1)
center1_k2 = np.uint8(center1_k2)
center1_k3 = np.uint8(center1_k3)

# تبدیل تصویر دوم به ازای سه مقدار کا
center2_k1 = np.uint8(center2_k1)
center2_k2 = np.uint8(center2_k2)
center2_k3 = np.uint8(center2_k3)

# تبدیل تصویر سوم به ازای سه مقدار کا
center3_k1 = np.uint8(center3_k1)
center3_k2 = np.uint8(center3_k2)
center3_k3 = np.uint8(center3_k3)
```

مرحله یازدهم : تبدیل ابعاد و ذخیره داده های تصاویر

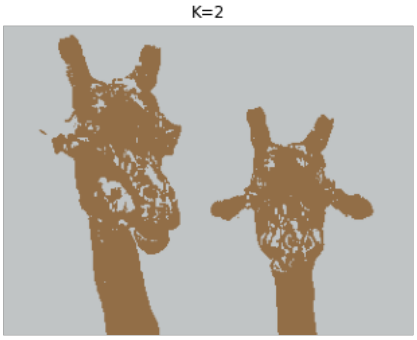
In [10]:

```
# convert all pixels to the color of the centroids  - تبدیل مرکز های خوشه بندی به رنگ بندی تصاویر
img1_k1 = center1_k1[label1_k1.flatten()]
img1_k2 = center1_k2[label1_k2.flatten()]
img1_k3 = center1_k3[label1_k3.flatten()]

img2_k1 = center2_k1[label2_k1.flatten()]
img2_k2 = center2_k2[label2_k2.flatten()]
img2_k3 = center2_k3[label2_k3.flatten()]

img3_k1 = center3_k1[label3_k1.flatten()]
img3_k2 = center3_k2[label3_k2.flatten()]
img3_k3 = center3_k3[label3_k3.flatten()]


# reshape back to the original image dimension  -  برگرداندن ابعاد تصاویر به ابعاد اصلی
img1_k1_2 = img1_k1.reshape((Orginal_img1.shape))
img1_k2_2 = img1_k2.reshape((Orginal_img1.shape))
img1_k3_2 = img1_k3.reshape((Orginal_img1.shape))

img2_k1_2 = img2_k1.reshape((Orginal_img2.shape))
img2_k2_2 = img2_k2.reshape((Orginal_img2.shape))
img2_k3_2 = img2_k3.reshape((Orginal_img2.shape))

img3_k1_2 = img3_k1.reshape((Orginal_img3.shape))
img3_k2_2 = img3_k2.reshape((Orginal_img3.shape))
img3_k3_2 = img3_k3.reshape((Orginal_img3.shape))
```

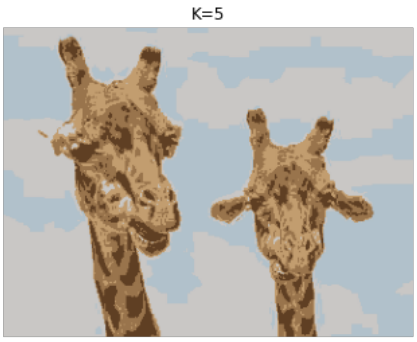
مرحله دوازدهم : نمایش نتایج خوشه بندی تصویر اول بهمراه نمودار مقادیر درون خوشه ای و برون خوشه ای

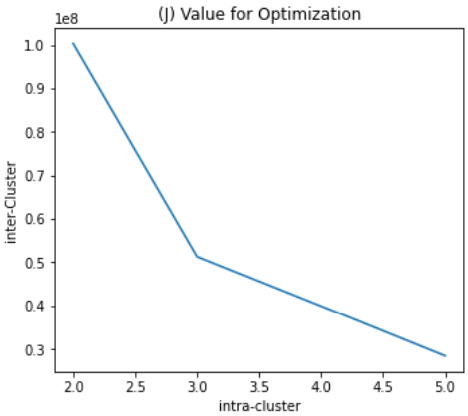
In [30]:

```
#  نمایش تصاویر خوشه بندی با میزان کا های مشخص شده برای تصویر اول
fig = plt.figure(figsize=(12, 16))
rows = 3
columns = 2
fig.add_subplot(rows, columns, 1)
plt.imshow(Orginal_img1)
plt.axis('off')
plt.title("Orginal")
fig.add_subplot(rows, columns, 2)
plt.imshow(img1_k1_2)
plt.axis('off')
plt.title("K=2")
fig.add_subplot(rows, columns, 3)
plt.imshow(img1_k2_2)
plt.axis('off')
plt.title("K=3")
fig.add_subplot(rows, columns, 4)
plt.imshow(img1_k3_2)
plt.axis('off')
plt.title("K=5")

# نمایش میزان تابع بهینه سازی برای خوشه بندی به ازای کا های مختلف
fig.add_subplot(rows, columns, 5)
# مقادیر درون خوشه ای
x = [cv.TERM_CRITERIA_MAX_ITER * k1,cv.TERM_CRITERIA_MAX_ITER * k2,cv.TERM_CRITERIA_MAX_ITER * k3]
# مقادیر تابع بهینه سازی خوشه سازی J
y=[0,0,0]
y[0] = ret1_k1 + cv.TERM_CRITERIA_EPS
y[1] = ret1_k2 + cv.TERM_CRITERIA_EPS
y[2] = ret1_k3 + cv.TERM_CRITERIA_EPS

plt.plot(x, y)
plt.xlabel('intra-cluster')
plt.ylabel('inter-Cluster')
plt.title('(J) Value for Optimization')
plt.show()
```



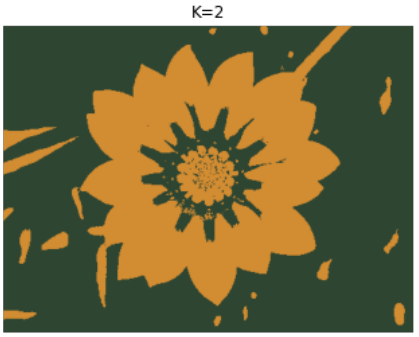


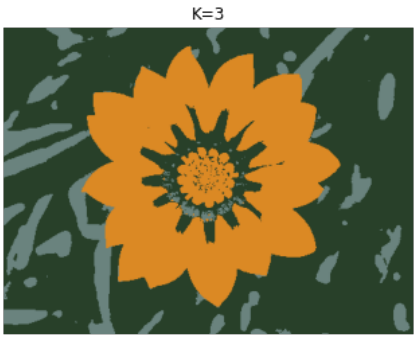



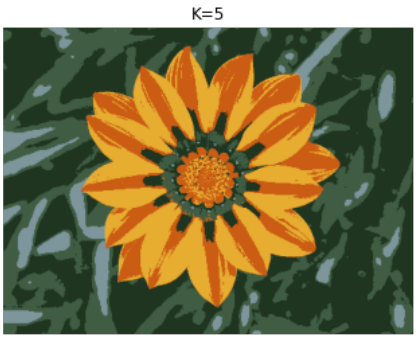

مرحله دوازدهم : نمایش نتایج خوشه بندی تصویر دوم بهمراه نمودار مقادیر درون خوشه ای و برون خوشه ای

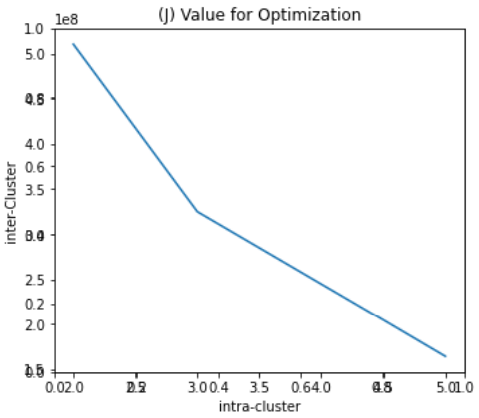
In [31]:

```
#  نمایش تصاویر خوشه بندی با میزان کا های مشخص شده برای تصویر دوم
fig = plt.figure(figsize=(12, 16))
rows = 3
columns = 2
fig.add_subplot(rows, columns, 1)
plt.imshow(Orginal_img2)
plt.axis('off')
plt.title("Orginal")
fig.add_subplot(rows, columns, 2)
plt.imshow(img2_k1_2)
plt.axis('off')
plt.title("K=2")
fig.add_subplot(rows, columns, 3)
plt.imshow(img2_k2_2)
plt.axis('off')
plt.title("K=3")
fig.add_subplot(rows, columns, 4)
plt.imshow(img2_k3_2)
plt.axis('off')
plt.title("K=5")

# نمایش میزان تابع بهینه سازی برای خوشه بندی به ازای کا های مختلف
fig.add_subplot(rows, columns, 5)
# نمایش میزان تابع بهینه سازی برای خوشه بندی به ازای کا های مختلف
fig.add_subplot(rows, columns, 5)
# مقادیر درون خوشه ای
x = [cv.TERM_CRITERIA_MAX_ITER * k1,cv.TERM_CRITERIA_MAX_ITER * k2,cv.TERM_CRITERIA_MAX_ITER * k3]
# مقادیر تابع بهینه سازی خوشه سازی J
y=[0,0,0]
y[0] = ret2_k1
y[1] = ret2_k2
y[2] = ret2_k3

plt.plot(x, y)
plt.xlabel('intra-cluster')
plt.ylabel('inter-Cluster')
plt.title('(J) Value for Optimization')
plt.show()
```

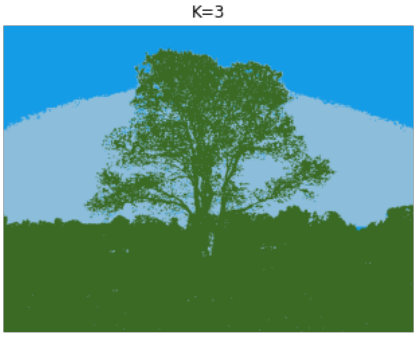




مرحله دوازدهم : نمایش نتایج خوشه بندی تصویر سوم بهمراه نمودار مقادیر درون خوشه ای و برون خوشه ای

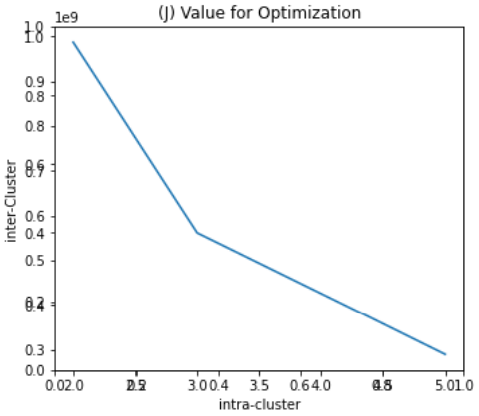
In [32]:

```
#  نمایش تصاویر خوشه بندی با میزان کا های مشخص شده برای تصویر دوم
fig = plt.figure(figsize=(12, 16))
rows = 3
columns = 2
fig.add_subplot(rows, columns, 1)
plt.imshow(Orginal_img3)
plt.axis('off')
plt.title("Orginal")
fig.add_subplot(rows, columns, 2)
plt.imshow(img3_k1_2)
plt.axis('off')
plt.title("K=2")
fig.add_subplot(rows, columns, 3)
plt.imshow(img3_k2_2)
plt.axis('off')
plt.title("K=3")
fig.add_subplot(rows, columns, 4)
plt.imshow(img3_k3_2)
plt.axis('off')
plt.title("K=5")

# نمایش میزان تابع بهینه سازی برای خوشه بندی به ازای کا های مختلف
fig.add_subplot(rows, columns, 5)
# نمایش میزان تابع بهینه سازی برای خوشه بندی به ازای کا های مختلف
fig.add_subplot(rows, columns, 5)
# مقادیر درون خوشه ای
x = [cv.TERM_CRITERIA_MAX_ITER * k1,cv.TERM_CRITERIA_MAX_ITER * k2,cv.TERM_CRITERIA_MAX_ITER * k3]
# مقادیر تابع بهینه سازی خوشه سازی J
y=[0,0,0]
y[0] = ret3_k1
y[1] = ret3_k2
y[2] = ret3_k3

plt.plot(x, y)
plt.xlabel('intra-cluster')
plt.ylabel('inter-Cluster')
plt.title('(J) Value for Optimization')
plt.show()
```










## جمع بندی نتایج پروژه

تصاویر را در متغیرهایی بارگذاری کردیم و سپس آن را به آرایه دو بعدی با رنگ بندی آر جی بی تغییر دادیم

سپس با استفاده از الگوریتم کی مین برای سه مقدار ۲ و ۳ و ۵ خوشه بندی نمودیم

سپس داده هایی که خوشه بندی کردیم را به تصاویر قابل مشاهده بازسازی یا تبدیل نمودیم

به ازای کای مشخص آن نشان دادیم همچنین در پایان نیز میزان تابع مینیمم برای کاهای مختلف طبق خوشه بندی الگوریتم کی مین بروی نمودار نشان دادیم

با تشکر فراوان - تهیه کننده : امین زایراومالی - دانشجو کارشناسی ارشد مهندسی کامپیوتر نرم افزار

استاد محترم : جناب دکتر آهنگران

دانشگاه علوم وفنون مازندارن