In [10]:

```
import numpy as np
import os
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
%matplotlib inline
from __future__ import print_function
from scipy.spatial.distance import cdist
np.random.seed(11)
```

Sử dụng thuật toán

In [24]:

```
means = [[2,2],[8,3],[3,6]]
cov = [[1,0],[0,1]]
N = 500
X0 = np.random.multivariate_normal(means[0], cov, N)
X1 = np.random.multivariate_normal(means[1], cov, N)
X2 = np.random.multivariate_normal(means[2], cov, N)

X = np.concatenate((X0,X1,X2), axis=0)
K=3

original_label = np.asarray([0]*N+[1]*N+[2]*N).T
```

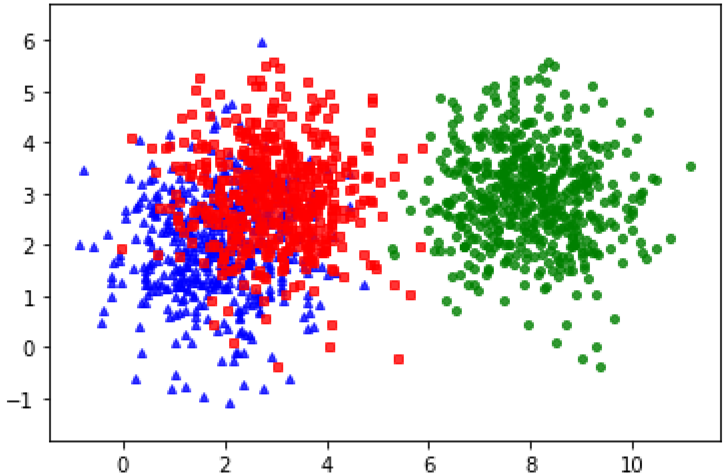
In [25]:

```
def kmeans_display(X,label):
    K = np.amax(label)+1
    X0=X[label==0,:]
    X1=X[label==1,:]
    X2=X[label==2,:]

    plt.plot(X0[:,0],X0[:,1],'b^',markersize=4,alpha=.8)
    plt.plot(X1[:,0],X1[:,1],'go',markersize=4,alpha=.8)
    plt.plot(X2[:,0],X1[:,1],'rs',markersize=4,alpha=.8)

    plt.axis('equal')
    plt.plot()
    plt.show()
```

```
kmeans_display(X, original_label)
```



In [23]:

```
def kmeans_init_centers(X,k):
    return X[np.random.choice(X.shape[0],k,replace=False)]
def kmeans_assign_labels(X,centers):
    D = cdist(X, centers)
    return np.argmin(D,axis=1)
def kmeans_update_centers(X,labels,K):
    centers = np.zeros((K,X.shape[1]))
    for k in range(K):
        Xk = X[labels==k,:]
        centers[k,:] = np.mean(Xk,axis=0)
    return centers
def has_converged(centers,new_centers):
    return (set([tuple(a) for a in centers]) ==
        set([tuple(a) for a in new_centers]))
```

Sử dụng thư viện

In [2]:

```
path = '25_Nguyen Van Linh_Ch3_K-means.csv'
df = pd.read_csv(path)
df.head(10)
```

Out[2]:

CustomerID	Gender	Age	Income	Spending_Score

	CustomerID	Gender	Age	Income	Spending_Score
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
5	6	Female	22	17	76
6	7	Female	35	18	6
7	8	Female	23	18	94
8	9	Male	64	19	3
9	10	Female	30	19	72

In [3]:

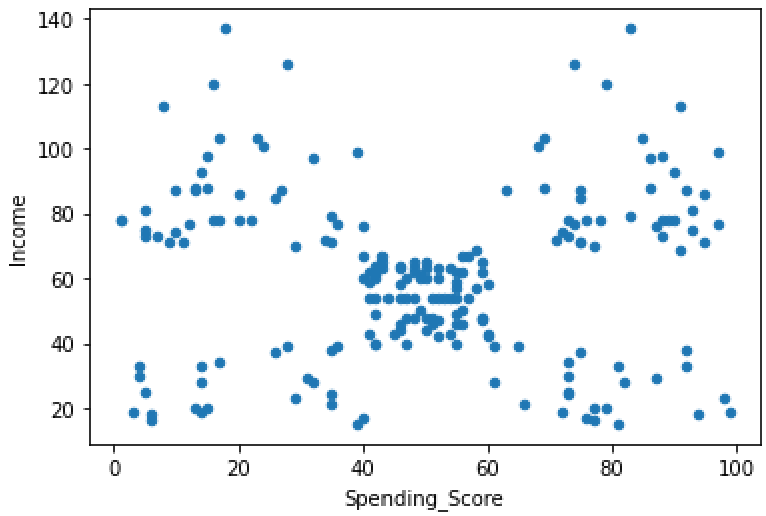
```
df.shape
```

Out[3]:

```
(200, 5)
```

In [4]:

```
df.plot(kind='scatter', x='Spending_Score', y='Income')
plt.show()
```



In [5]:

```
df = df.drop(['CustomerID', 'Gender', 'Age'], axis=1)
df
```

Out[5]:

	Income	Spending_Score
0	15	39
1	15	81
2	16	6
3	16	77
4	17	40
...	...	...
195	120	79
196	126	28
197	126	74
198	137	18
199	137	83

200 rows × 2 columns

In [6]:
```
df.corr()
```

Out[6]:

	Income	Spending_Score
Income	1.000000	0.009903
Spending_Score	0.009903	1.000000

In [ ]:
```
customcmap = ListedColormap(["crimson", "mediumblue", "darkmagenta"])

fig, ax = plt.subplots(figsize=(8, 6))
plt.scatter(x=blobs['x'], y=blobs['y'], s=150,
            c=blobs['cluster'].astype('category'),
            cmap = customcmap)
ax.set_xlabel(r'x', fontsize=14)
ax.set_ylabel(r'y', fontsize=14)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.show()
```

In [6]:
```
sc = StandardScaler()
data_stand = sc.fit_transform(df)
data_stand
```

Out[6]:
```
array([[-1.73899919, -0.43480148],
       [-1.73899919,  1.19570407],
       [-1.70082976, -1.71591298],
       [-1.70082976,  1.04041783],
       [-1.66266033, -0.39597992],
       [-1.66266033,  1.00159627],
       [-1.62449091, -1.71591298],
       [-1.62449091,  1.70038436],
       [-1.58632148, -1.83237767],
       [-1.58632148,  0.84631002],
       [-1.58632148, -1.4053405 ],
       [-1.58632148,  1.89449216],
       [-1.54815205, -1.36651894],
       [-1.54815205,  1.04041783],
       [-1.54815205, -1.44416206],
       [-1.54815205,  1.11806095],
       [-1.50998262, -0.59008772],
       [-1.50998262,  0.61338066],
```

```
[-1.43364376, -0.82301709],
[-1.43364376,  1.8556706 ],
[-1.39547433, -0.59008772],
[-1.39547433,  0.88513158],
[-1.3573049 , -1.75473454],
[-1.3573049 ,  0.88513158],
[-1.24279661, -1.4053405 ],
[-1.24279661,  1.23452563],
[-1.24279661, -0.7065524 ],
[-1.24279661,  0.41927286],
[-1.20462718, -0.74537397],
[-1.20462718,  1.42863343],
[-1.16645776, -1.7935561 ],
[-1.16645776,  0.88513158],
[-1.05194947, -1.7935561 ],
[-1.05194947,  1.62274124],
[-1.05194947, -1.4053405 ],
[-1.05194947,  1.19570407],
[-1.01378004, -1.28887582],
[-1.01378004,  0.88513158],
[-0.89927175, -0.93948177],
[-0.89927175,  0.96277471],
[-0.86110232, -0.59008772],
[-0.86110232,  1.62274124],
[-0.82293289, -0.55126616],
[-0.82293289,  0.41927286],
[-0.82293289, -0.86183865],
[-0.82293289,  0.5745591 ],
[-0.78476346,  0.18634349],
[-0.78476346, -0.12422899],
[-0.78476346, -0.3183368 ],
[-0.78476346, -0.3183368 ],
[-0.70842461,  0.06987881],
[-0.70842461,  0.38045129],
[-0.67025518,  0.14752193],
[-0.67025518,  0.38045129],
[-0.67025518, -0.20187212],
[-0.67025518, -0.35715836],
[-0.63208575, -0.00776431],
[-0.63208575, -0.16305055],
[-0.55574689,  0.03105725],
[-0.55574689, -0.16305055],
[-0.55574689,  0.22516505],
[-0.55574689,  0.18634349],
[-0.51757746,  0.06987881],
```

```
[-0.51757746,  0.34162973],
[-0.47940803,  0.03105725],
[-0.47940803,  0.34162973],
[-0.47940803, -0.00776431],
[-0.47940803, -0.08540743],
[-0.47940803,  0.34162973],
[-0.47940803, -0.12422899],
[-0.4412386 ,  0.18634349],
[-0.4412386 , -0.3183368 ],
[-0.40306917, -0.04658587],
[-0.40306917,  0.22516505],
[-0.25039146, -0.12422899],
[-0.25039146,  0.14752193],
[-0.25039146,  0.10870037],
[-0.25039146, -0.08540743],
[-0.25039146,  0.06987881],
[-0.25039146, -0.3183368 ],
[-0.25039146,  0.03105725],
[-0.25039146,  0.18634349],
[-0.25039146, -0.35715836],
[-0.25039146, -0.24069368],
[-0.25039146,  0.26398661],
[-0.25039146, -0.16305055],
[-0.13588317,  0.30280817],
[-0.13588317,  0.18634349],
[-0.09771374,  0.38045129],
[-0.09771374, -0.16305055],
[-0.05954431,  0.18634349],
[-0.05954431, -0.35715836],
[-0.02137488, -0.04658587],
[-0.02137488, -0.39597992],
[-0.02137488, -0.3183368 ],
[-0.02137488,  0.06987881],
[-0.02137488, -0.12422899],
[-0.02137488, -0.00776431],
[ 0.01679455, -0.3183368 ],
[ 0.01679455, -0.04658587],
[ 0.05496398, -0.35715836],
[ 0.05496398, -0.08540743],
[ 0.05496398,  0.34162973],
[ 0.05496398,  0.18634349],
[ 0.05496398,  0.22516505],
[ 0.05496398, -0.3183368 ],
[ 0.09313341, -0.00776431],
[ 0.09313341, -0.16305055],
```

```
[ 0.09313341, -0.27951524],
[ 0.09313341, -0.08540743],
[ 0.09313341,  0.06987881],
[ 0.09313341,  0.14752193],
[ 0.13130284, -0.3183368 ],
[ 0.13130284, -0.16305055],
[ 0.16947227, -0.08540743],
[ 0.16947227, -0.00776431],
[ 0.16947227, -0.27951524],
[ 0.16947227,  0.34162973],
[ 0.24581112, -0.27951524],
[ 0.24581112,  0.26398661],
[ 0.24581112,  0.22516505],
[ 0.24581112, -0.39597992],
[ 0.32214998,  0.30280817],
[ 0.32214998,  1.58391968],
[ 0.36031941, -0.82301709],
[ 0.36031941,  1.04041783],
[ 0.39848884, -0.59008772],
[ 0.39848884,  1.73920592],
[ 0.39848884, -1.52180518],
[ 0.39848884,  0.96277471],
[ 0.39848884, -1.5994483 ],
[ 0.39848884,  0.96277471],
[ 0.43665827, -0.62890928],
[ 0.43665827,  0.80748846],
[ 0.4748277 , -1.75473454],
[ 0.4748277 ,  1.46745499],
[ 0.4748277 , -1.67709142],
[ 0.4748277 ,  0.88513158],
[ 0.51299713, -1.56062674],
[ 0.51299713,  0.84631002],
[ 0.55116656, -1.75473454],
[ 0.55116656,  1.6615628 ],
[ 0.58933599, -0.39597992],
[ 0.58933599,  1.42863343],
[ 0.62750542, -1.48298362],
[ 0.62750542,  1.81684904],
[ 0.62750542, -0.55126616],
[ 0.62750542,  0.92395314],
[ 0.66567484, -1.09476801],
[ 0.66567484,  1.54509812],
[ 0.66567484, -1.28887582],
[ 0.66567484,  1.46745499],
[ 0.66567484, -1.17241113],
```

```
[ 0.66567484,  1.00159627],
[ 0.66567484, -1.32769738],
[ 0.66567484,  1.50627656],
[ 0.66567484, -1.91002079],
[ 0.66567484,  1.07923939],
[ 0.66567484, -1.91002079],
[ 0.66567484,  0.88513158],
[ 0.70384427, -0.59008772],
[ 0.70384427,  1.27334719],
[ 0.78018313, -1.75473454],
[ 0.78018313,  1.6615628 ],
[ 0.93286085, -0.93948177],
[ 0.93286085,  0.96277471],
[ 0.97103028, -1.17241113],
[ 0.97103028,  1.73920592],
[ 1.00919971, -0.90066021],
[ 1.00919971,  0.49691598],
[ 1.00919971, -1.44416206],
[ 1.00919971,  0.96277471],
[ 1.00919971, -1.56062674],
[ 1.00919971,  1.62274124],
[ 1.04736914, -1.44416206],
[ 1.04736914,  1.38981187],
[ 1.04736914, -1.36651894],
[ 1.04736914,  0.72984534],
[ 1.23821628, -1.4053405 ],
[ 1.23821628,  1.54509812],
[ 1.390894  , -0.7065524 ],
[ 1.390894  ,  1.38981187],
[ 1.42906343, -1.36651894],
[ 1.42906343,  1.46745499],
[ 1.46723286, -0.43480148],
[ 1.46723286,  1.81684904],
[ 1.54357172, -1.01712489],
[ 1.54357172,  0.69102378],
[ 1.61991057, -1.28887582],
[ 1.61991057,  1.35099031],
[ 1.61991057, -1.05594645],
[ 1.61991057,  0.72984534],
[ 2.00160487, -1.63826986],
[ 2.00160487,  1.58391968],
[ 2.26879087, -1.32769738],
[ 2.26879087,  1.11806095],
[ 2.49780745, -0.86183865],
[ 2.49780745,  0.92395314],
```

```
[ 2.91767117, -1.25005425],
[ 2.91767117,  1.27334719]])
```

In [7]:

```python
data_stand = pd.DataFrame(data_stand, columns=['Spending_Score', 'Income'])
data_stand
```

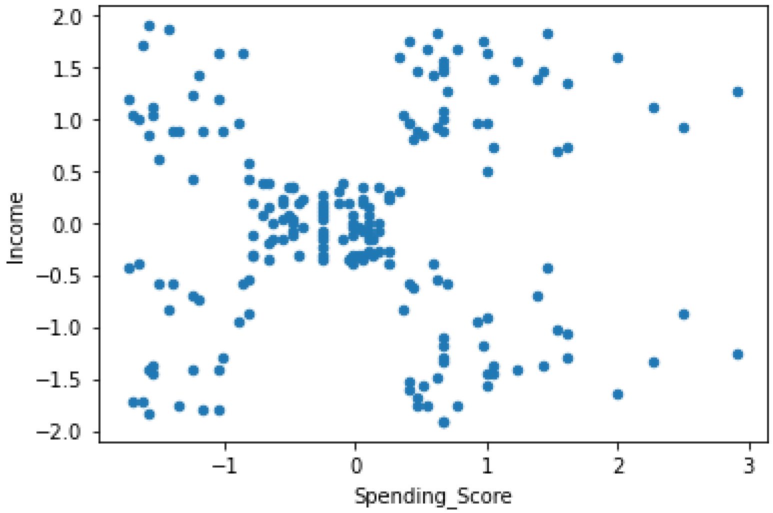
Out[7]:

	Spending_Score	Income
0	-1.738999	-0.434801
1	-1.738999	1.195704
2	-1.700830	-1.715913
3	-1.700830	1.040418
4	-1.662660	-0.395980
...	...	...
195	2.268791	1.118061
196	2.497807	-0.861839
197	2.497807	0.923953
198	2.917671	-1.250054
199	2.917671	1.273347

200 rows × 2 columns

In [8]:

```python
data_stand.plot(kind='scatter', x='Spending_Score', y='Income')
plt.show()
```



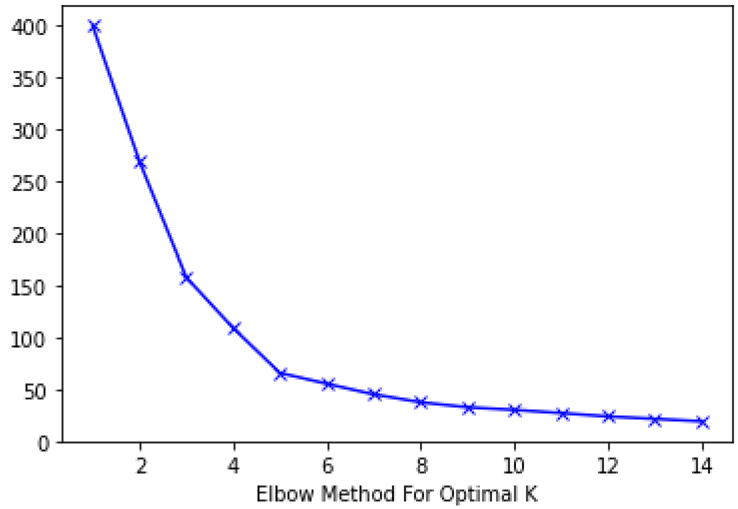
In [67]:

```
Sum_of_squared_distances = []
K = range(1,15)
for k in K:
    km=KMeans(n_clusters=k)
    km=km.fit(data_stand)
    Sum_of_squared_distances.append(km.inertia_)
```

```
C:\Users\fna\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:881: UserWarning: KMeans is known to have a memory le
ak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment va
riable OMP_NUM_THREADS=1.
  warnings.warn(
```

In [10]:

```
plt.plot(K, Sum_of_squared_distances, 'bx-')
plt.xlabel('k')
plt.xlabel('Sum_of_squared_distances')
plt.xlabel('Elbow Method For Optimal K')
plt.show()
```



K=5

In [11]:

```
km5 = KMeans(n_clusters = 5)
km5 = km5.fit(df)
print(km5.labels_)
```

```
[3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3
 0 3 0 3 0 3 4 3 0 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
 4 4 4 4 4 4 4 4 4 4 4 4 2 1 2 4 2 1 2 1 2 4 2 1 2 1 2 1 2 1 2 4 2 1 2 1 2
 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1
 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2]
```

In [62]:

```
labels_1 = km5.labels_
labels_1 = pd.DataFrame(labels_1, columns=['Cluster'])
df_clustered_1 = pd.concat([df,labels_1], axis=1)
```

In [63]:

```
df_clustered_1.sort_values('Cluster', ascending=True)
```

Out[63]:

	Income	Spending_Score	Cluster
41	38	92	0
23	25	73	0

	Income	Spending_Score	Cluster
29	29	87	0
21	24	73	0
39	37	75	0
...	...	...	...
126	71	35	4
92	60	49	4
93	60	40	4
85	54	46	4
99	61	49	4

200 rows × 3 columns

In [64]:

```
df_clustered_1[df_clustered_1['Cluster']==0]
```

Out[64]:

	Income	Spending_Score	Cluster
1	15	81	0
3	16	77	0
5	17	76	0
7	18	94	0
9	19	72	0
11	19	99	0
13	20	77	0
15	20	79	0
17	21	66	0
19	23	98	0
21	24	73	0

	Income	Spending_Score	Cluster
23	25	73	0
25	28	82	0
27	28	61	0
29	29	87	0
31	30	73	0
33	33	92	0
35	33	81	0
37	34	73	0
39	37	75	0
41	38	92	0
45	39	65	0

In [65]:

```
centroids_1 = km5.cluster_centers_
centroids_1 = pd.DataFrame(centroids_1, columns=['Centroid_SpendingScore', 'Centroid_Income'])
centroids_1
```

Out[65]:

	Centroid_SpendingScore	Centroid_Income
0	25.727273	79.363636
1	88.200000	17.114286
2	86.538462	82.128205
3	26.304348	20.913043
4	55.296296	49.518519

In [66]:

```
s1 = sns.scatterplot(data=df_clustered_1, x='Spending_Score', y='Income', hue='Cluster')
centroids.plot(ax=s1, kind='scatter', x='Centroid_SpendingScore', y='Centroid_Income', color='red')
```

Out[66]:

```
<AxesSubplot:xlabel='Centroid_SpendingScore', ylabel='Centroid_Income'>
```

K=3

In [56]:
```
km3 = KMeans(n_clusters = 3)
km3 = km3.fit(df)
print(km3.labels_)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2
 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1
 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2]
```

In [57]:
```
labels_2 = km3.labels_
labels_2 = pd.DataFrame(labels, columns=['Cluster'])
df_clustered_2 = pd.concat([df,labels_2], axis=1)
```

In [58]:
```
df_clustered_2.sort_values('Cluster', ascending=True)
```

Out[58]:

	Income	Spending_Score	Cluster
0	15	39	0
90	59	55	0

	Income	Spending_Score	Cluster
89	58	46	0
88	58	60	0
87	57	55	0
...	...	...	...
135	73	88	2
173	87	92	2
147	77	74	2
133	72	71	2
199	137	83	2

200 rows × 3 columns

In [59]:

```
centroids_2 = km3.cluster_centers_
centroids_2 = pd.DataFrame(centroids_2, columns=['Centroid_SpendingScore', 'Centroid_Income'])
centroids_2
```

Out[59]:

	Centroid_SpendingScore	Centroid_Income
0	44.154472	49.829268
1	87.000000	18.631579
2	86.538462	82.128205

In [61]:

```
s2 = sns.scatterplot(data=df_clustered_2, x='Spending_Score', y='Income', hue='Cluster')
centroids_2.plot(ax=s2, kind='scatter', x='Centroid_SpendingScore', y='Centroid_Income', color='red')
```

Out[61]:

```
<AxesSubplot:xlabel='Centroid_SpendingScore', ylabel='Centroid_Income'>
```


Cluster
0
1
2
Centroid_Income
140
120
100
80
60
40
20
0
20
40
60
80
100
Centroid_SpendingScore