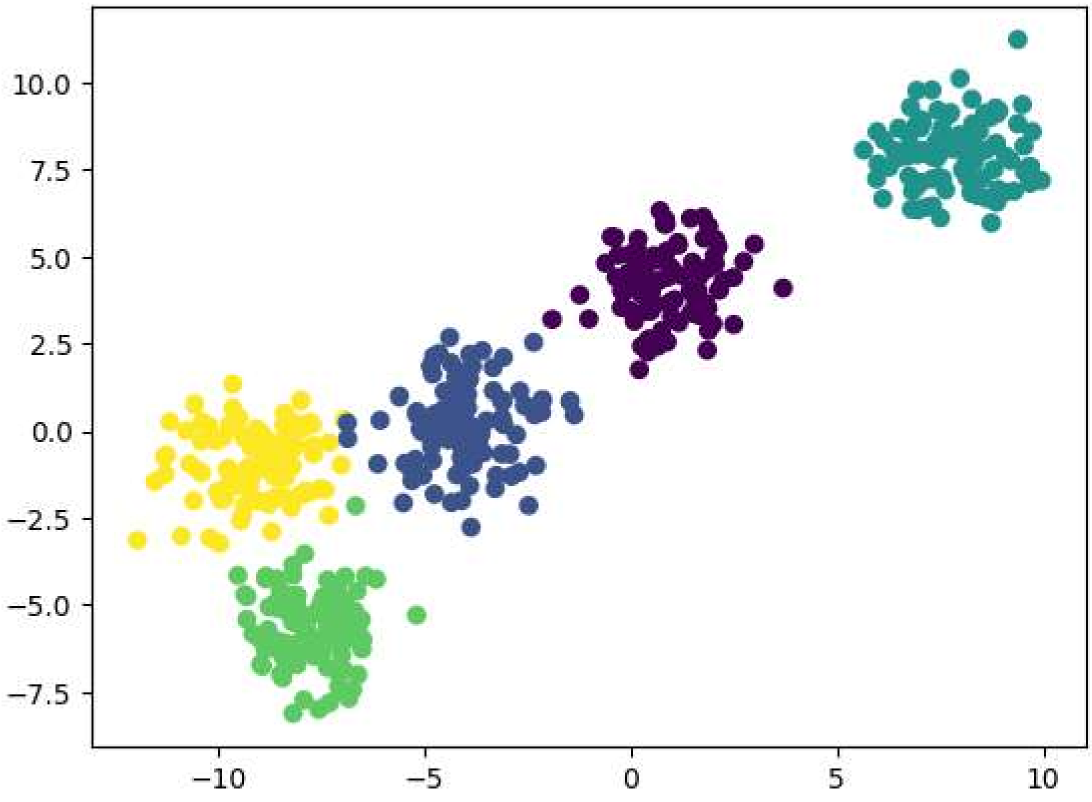
2.K-Means

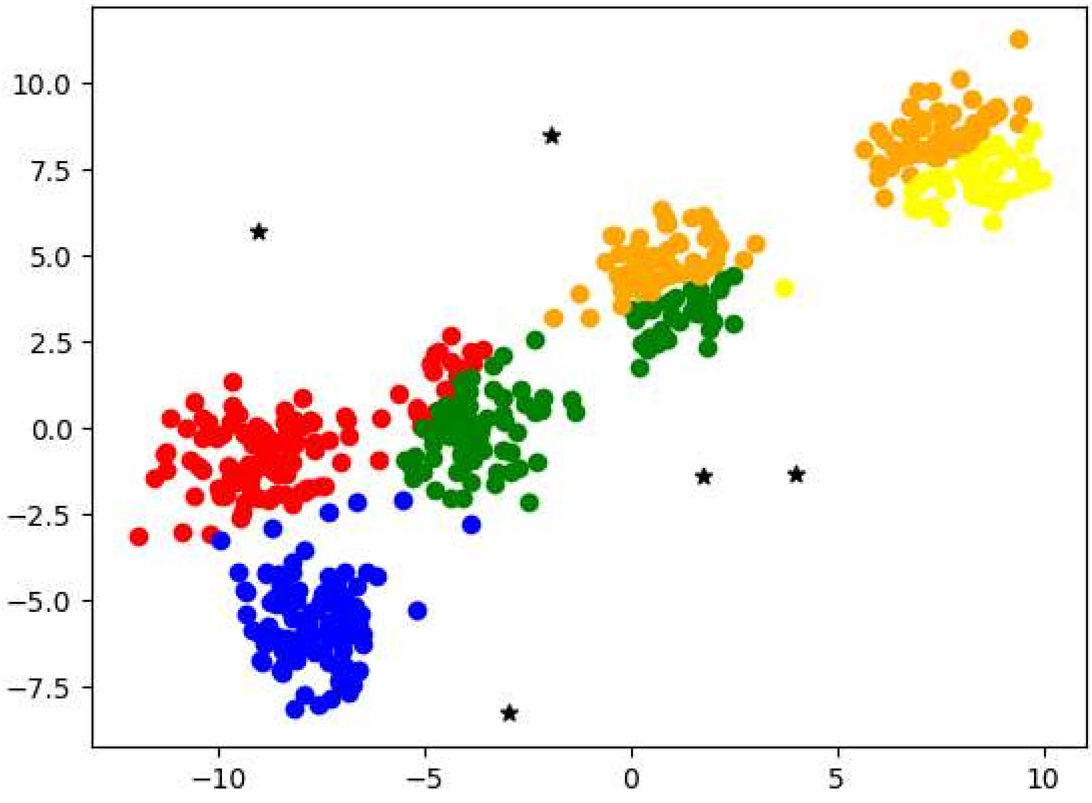
In [ ]:
```
import numpy as np
from matplotlib import pyplot as plt
from sklearn.datasets import make_blobs

X, y = make_blobs(n_samples=500, n_features=2, centers=5, random_state=3)

plt.figure(0)
plt.scatter(X[:, 0], X[:, 1], c=y)
plt.show()
```

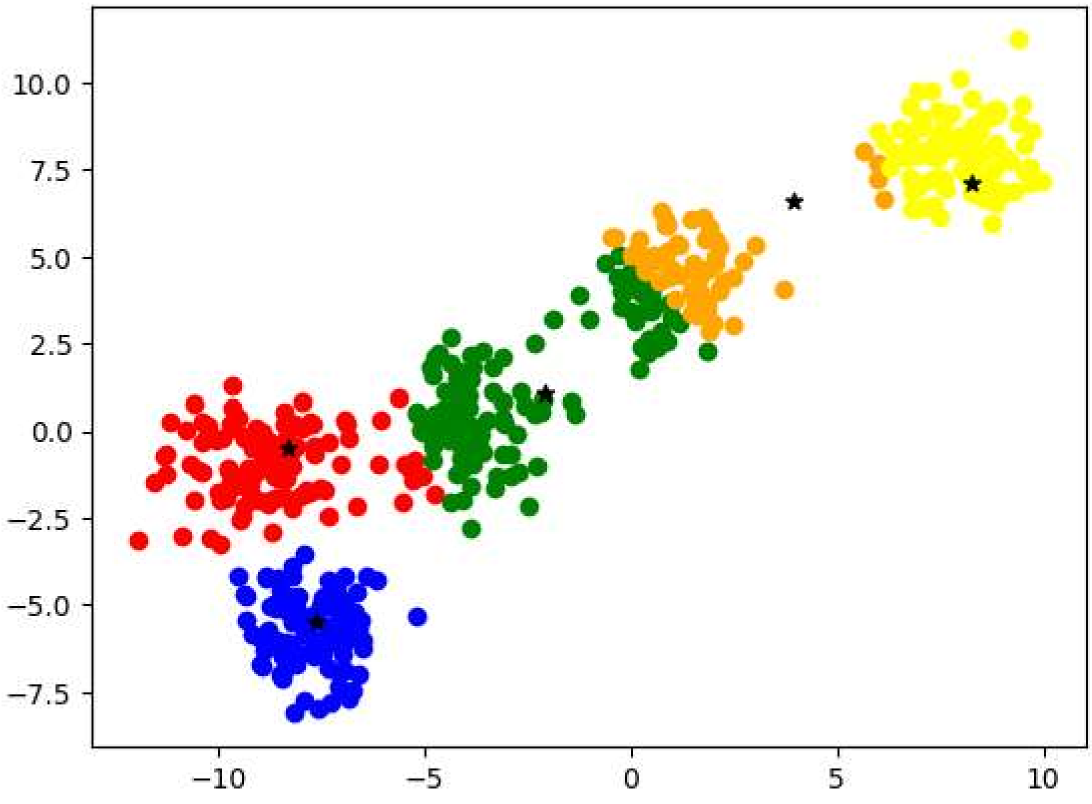


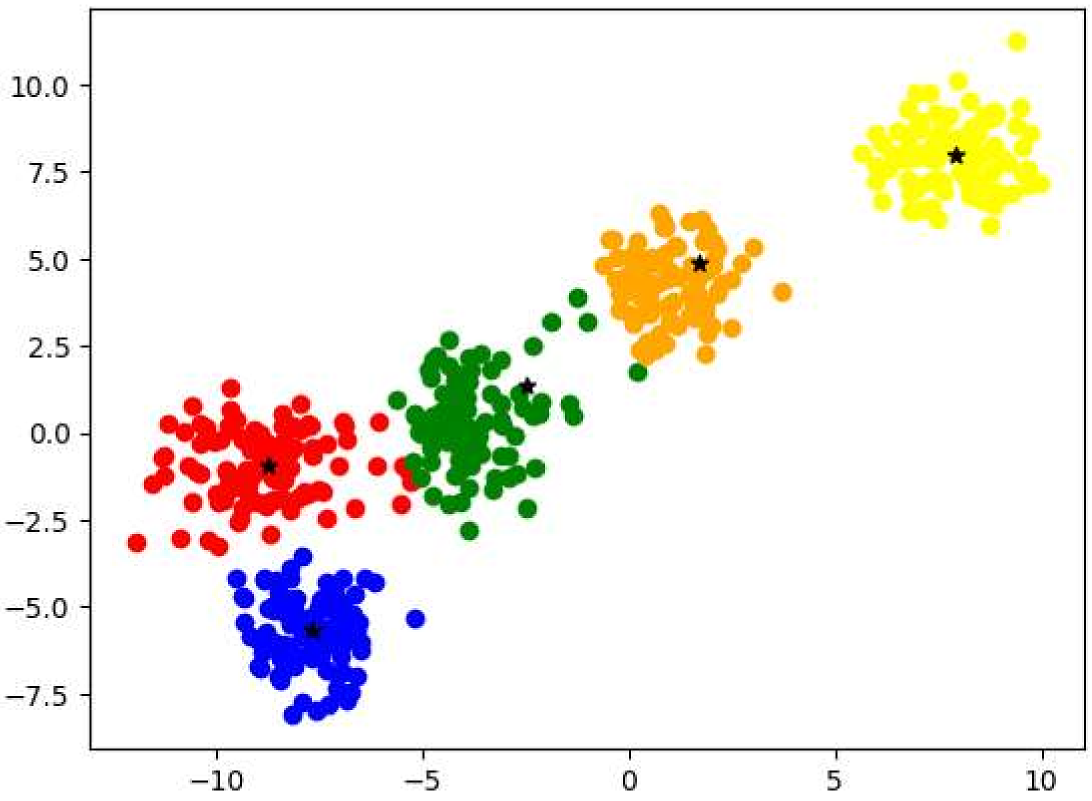
In [ ]:
```
k = 5
color = ["red", "green", "blue", "orange", "yellow"]

clusters = {}

for i in range(k):
    center = 10*(2*np.random.random((X.shape[1],))-1)
    pts = []
    cluster = {
        "center": center,
        "pts": pts,
        "color": color[i]
    }

    clusters[i] = cluster
```

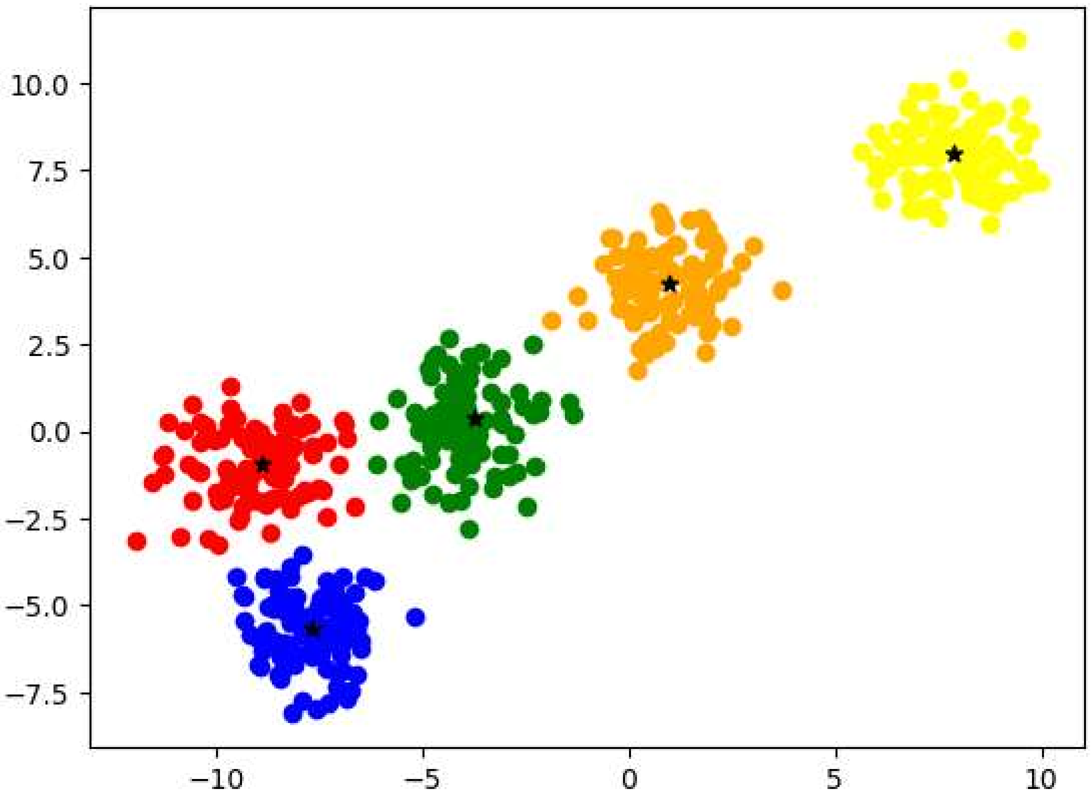
In [ ]:
```
def distance(p1, p2):
    return np.sqrt(np.sum((p1 - p2)**2))

def assignPointToCluster(clusters):
    for x in range(X.shape[0]):
        dist = []
        curr_point = X[x]

        for i in range(k):
            curr_dist = distance(clusters[i]['center'], curr_point)
            dist.append(curr_dist)

        curr_cluster = np.argmin(dist)
        clusters[curr_cluster]['pts'].append(curr_point)

def updateClusters(clusters):
    for kx in range(k):
        pts = np.array(clusters[kx]['pts'])

        if pts.shape[0] > 0:
            new_u = pts.mean(axis=0)
            clusters[kx]['center'] = new_u
            clusters[kx]['pts'] = []


def plotClusters(clusters):
    plt.figure()
    for kx in range(k):
        pts = np.array(clusters[kx]['pts'])

        try:
            plt.scatter(pts[:, 0], pts[:, 1], color = clusters[kx]['color'])
        except:
            pass

        center = clusters[kx]['center']
        plt.scatter(center[0], center[1], color="black", marker='*')
```

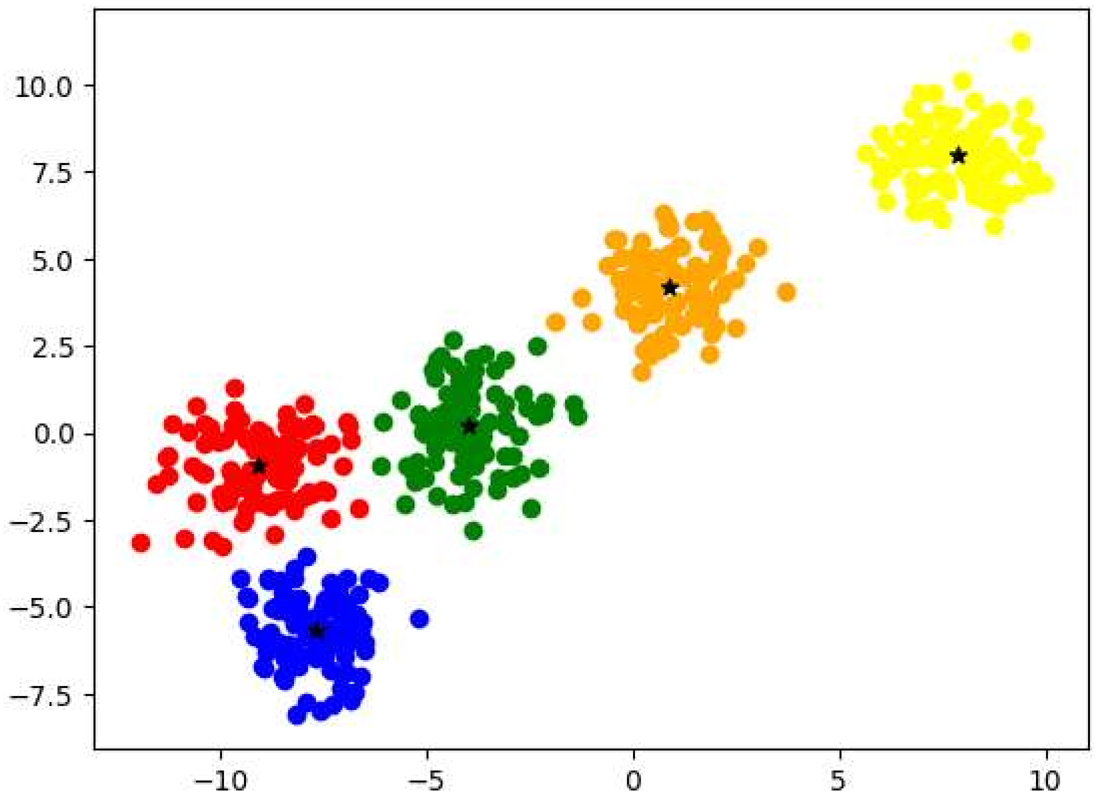
```
In [ ]: epoch = 5
        for i in range(epoch):
            assignPointToCluster(clusters)
            plotClusters(clusters)
            updateClusters(clusters)
```






10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10
−5
0
5
10


10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10
−5
0
5
10



```
In [ ]:
```

```
In [ ]:
```