Part A

a.Provide your algorithm
Code is in ./Hw3/kMeansAlgo/kMeans.py

Algorithm :

1. Select random k data points as centroids
2. Calculate the distance of all the data points from these centroids
3. Assign the points to centroids they are closest to
4. average the cluster data points to re-calculate the centroids
5. If the centroids don't move much, go to step 6. Else, go to step 1
6. Scatter plot the data points with the centroids they belong to

-

b.Provide your test cases including boundary test conditions and the clusters you obtained on applying your algorithm

TEST CASES :

Dataset 1: [[2,4],[2,6], [2,8], [10,4], [10,6], [10,8]]

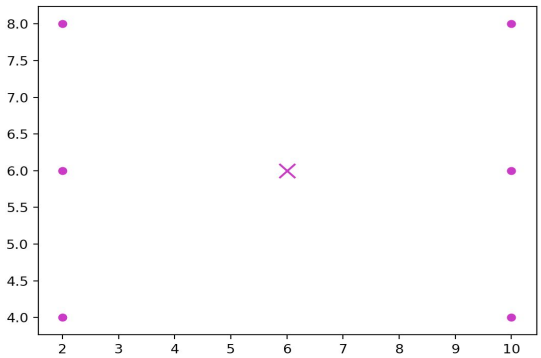
Tested with values k = 0,1,2,3,6,7

Results :

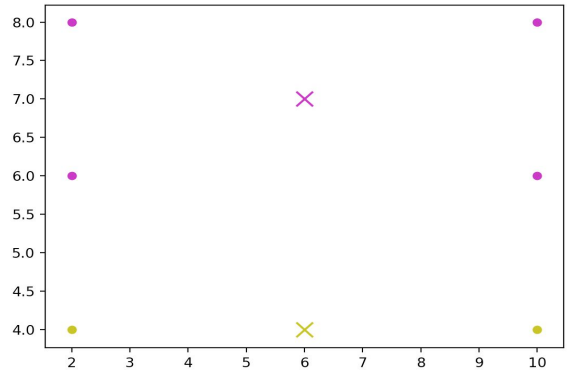
K = 0
I have added an error for this case as number of clusters cannot be zero

K = 7
Added error for this as it does not make sense to have more number of clusters than the data points

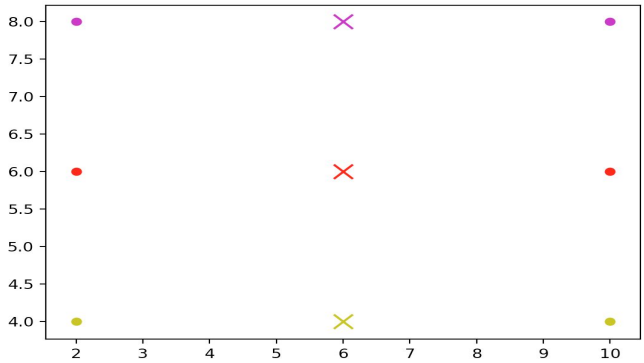
K = 1,



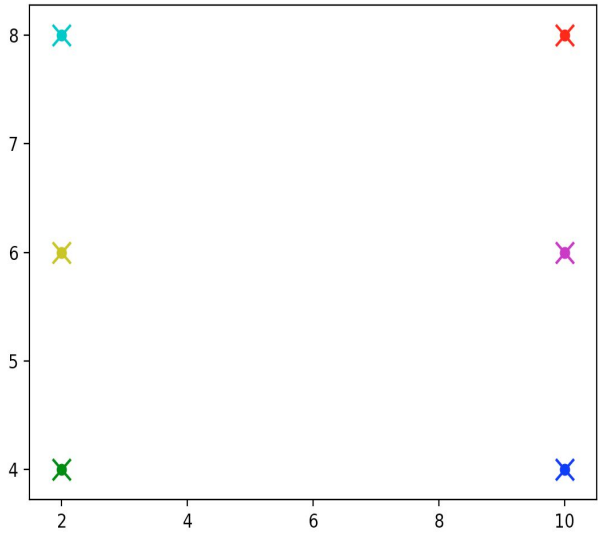
K = 2,



K = 3,



K = 6



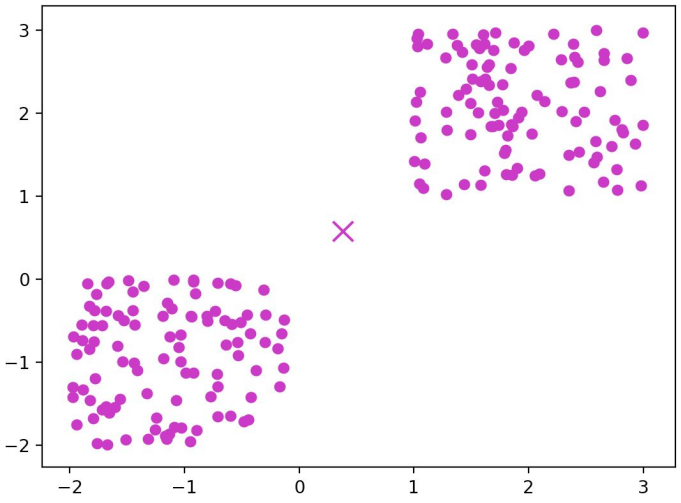
I have also added error for empty dataset, as it does not make sense to apply the algorithm in case of empty dataset.

Dataset 2 :

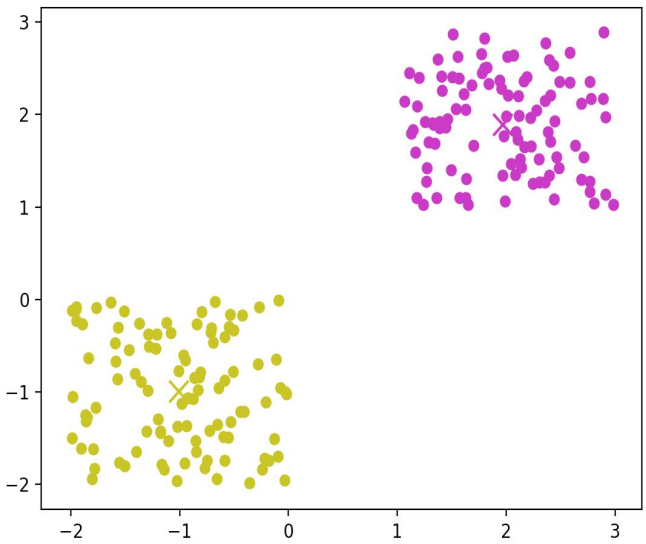
```
X= -2 * np.random.rand(200,2)
X1 = 1 + 2 * np.random.rand(100,2)
X[100:200, :] = X1
```

Generated random data

K = 1



K =2



c.Mention the drawbacks of the k-means algorithm

- Difficult to predict K-Value.
- Different initialization values of centroids give different results
- Can be handled for numerical data only