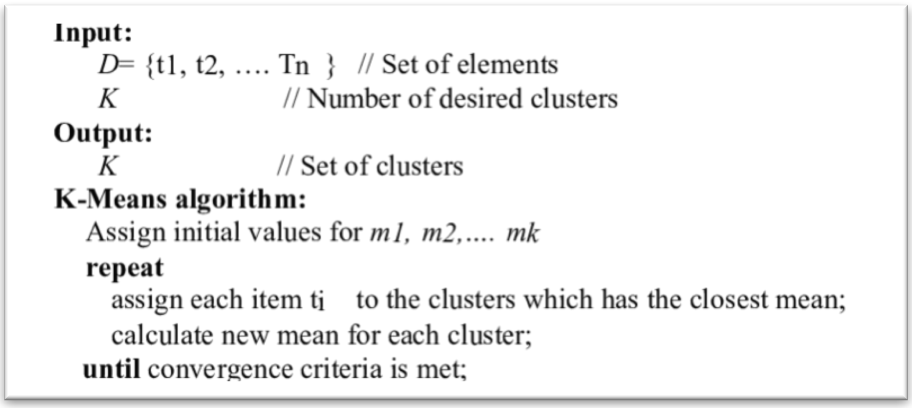
Implementation 1: KMeans

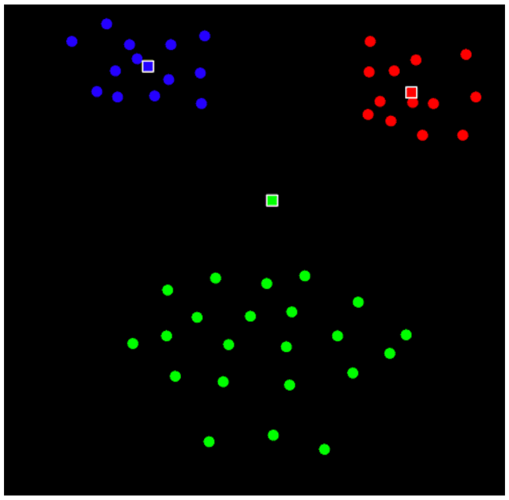
In this part, you will implement and use the K-means clustering algorithm. First you will learn to cluster a simple 2D dataset, next you will learn a method to evaluate the performance of clustering and finally you will learn about the restrictions of KMeans by running it on a complex dataset.

As discussed in the class, the main idea behind KMeans is an iterative process that starts by guessing the initial cluster centers, and then improves this guess by repeatedly assigning data points to their closest cluster center and then recalculating the centers based on the assignment. The pseudo-code of K-means is as follows:



```
Input:
    D= {t1, t2, .... Tn }   // Set of elements
    K                      // Number of desired clusters
Output:
    K                      // Set of clusters
K-Means algorithm:
   Assign initial values for m1, m2,.... mk
   repeat
      assign each item ti   to the clusters which has the closest mean;
      calculate new mean for each cluster;
   until convergence criteria is met;
```

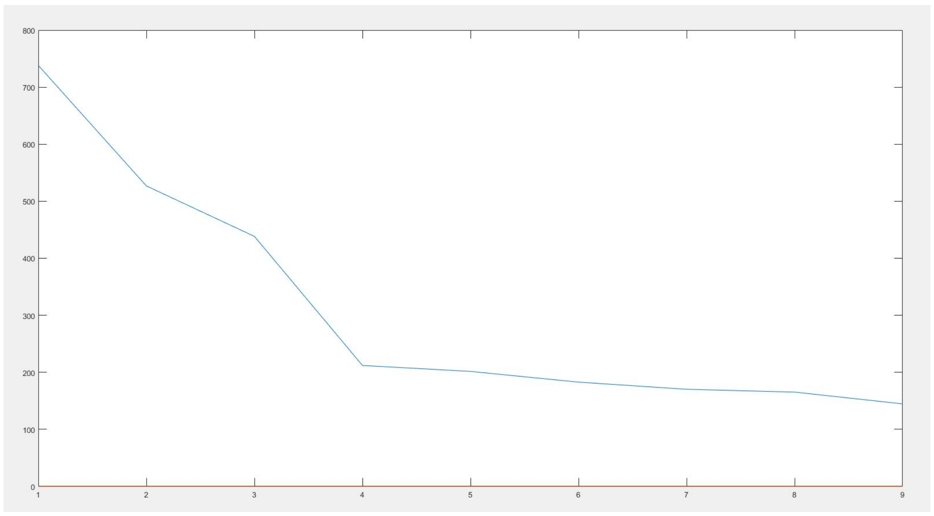
A) After completing the algorithm, run it on Dataset1. Set number of iteration to at least 15 and run K-means with k=2, 3, 4. After each run plot the clustered data points with each cluster having a different color. (you can use matplotlib to visualize the data)



B) After the clustering is done, compute for each cluster, the average distance between the cluster center and the data points in that cluster (this average distance is called cluster error).

C) compute the average cluster error and report it as the clustering error.

D) run the k-means with 0<k<15 on Dataset1 and compute the clustering error and plot these errors.



E) Use the "elbow" algorithm to find the optimum K.

F) Run the clustering once again on Dataset2, explain why KMeans fails on this dataset.