Semester	T.E. Semester V – Computer Engineering
Subject	Data Warehousing and Mining
Subject Professor In-charge	Prof. Kavita Shirsat
Assisting Teachers	Prof. Kavita Shirsat
Laboratory	M-313A

Student Name	Vibodh Bhosure	
Roll Number	20102A0032	
Grade and Subject Teacher's Signature		

Experiment Number	05	
Experiment Title	To implement K-Means Algorithm and find k clusters for a given value of k	
Resources / Apparatus Required	Hardware: Computer system	Software: Python
Description	K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on. It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training. It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters. The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm. The k-means clustering algorithm mainly performs two tasks:	

1. Determines the best value for K center points or centroids by an iterative process.
2. Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

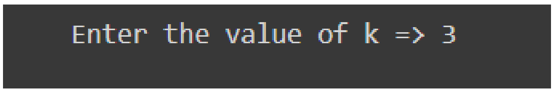
Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

Program

```
# -*- coding: utf-8 -*-
"""K-Means.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/10QpBOA
Irct8pThb1RYCy_bg8rBdDarXo

# Code By - Vibodh Bhosure
"""

import pandas as pd
import numpy as np
import statistics as st
import random

inp = [22,9,12,15,10,27,35,18,36,11]

k = int(input("Enter the value of k => "))

randomValue = []
for i in range(0, k):
  rand = inp[random.randint(0, len(inp)-1)]
  if rand not in randomValue:
    randomValue.append(rand)
print(randomValue)

def distance(a,b):
  return abs(a-b)

def itr(randomValue):
  global c1
  global c2
  global c3
  c1=[]
  c2=[]
  c3=[]
  lst = []
  for i in inp:
    d1 = distance(i,randomValue[0])
    d2 = distance(i,randomValue[1])
    d3 = distance(i,randomValue[2])
    val = {"d1":d1,"d2":d2,"d3":d3}
    minValue = sorted(val.items(), key=lambda
t: t[1])[0][0]
```

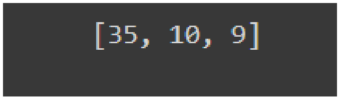
```
    if minValue == "d1":
      c1.append(i)
    elif minValue == "d2":
      c2.append(i)
    else: c3.append(i)
  # print(c1)
  # print(c2)
  # print(c3)
  c1 = np.array(c1)
  c2 = np.array(c2)
  c3 = np.array(c3)
  c1_mean = np.mean(c1)
  c2_mean = np.mean(c2)
  c3_mean = np.mean(c3)
  c1 = list(c1)
  c2 = list(c2)
  c3 = list(c3)
  lst.append(c1_mean)
  lst.append(c2_mean)
  lst.append(c3_mean)
  # print(c1_mean,c2_mean,c3_mean)
  return lst

prev_lst = []
new_lst = itr(randomValue)
while prev_lst != new_lst:
  prev_lst = new_lst
  new_lst = itr(new_lst)

print(c1, c2, c3)
```

Output



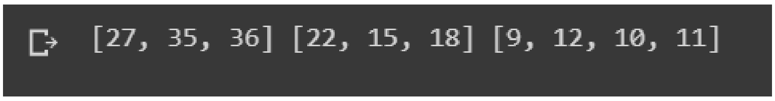
```
Enter the value of k => 3
```

Random centroids(means)-



```
[35, 10, 9]
```

Three clusters-



```
[27, 35, 36] [22, 15, 18] [9, 12, 10, 11]
```

Conclusion:

Hence, a random dataset was taken and considering k=3, three clusters were obtained. For different values of centroid(mean), the same clusters were obtained, hence, it shows that the obtained output is correct.