```
#!/usr/bin/python3
import pandas as pd
import random
import numpy as np
import sys
import os

class KMEANS(object):
    import numpy as np
    def __init__(self, k, metricOrder):
        self.k=k
        self.ord=metricOrder
        self.maxIterations=100

    def setMaxIter(self,maxIterations):
        self.maxIterations=maxIterations

    def getDB(self, DB):
        self.DB=DB
        self.labelCentroids={}
        self.iterations=0
        self.oldCentroids=np.empty(shape=(self.k,self.DB.shape[1]))
        self.labelData=np.concatenate((np.ndarray((self.DB.shape[0],1)),self.DB), axis=1) #App
end cluster id 0 to all data

    def fit(self, DB):
        self.getDB(DB)
        self.initCentroids() #Initialize centroids randomly from available dataset
        while(not self.shouldStop()):
            self.oldCentroids=np.copy(self.centroids)
#             print(self.oldCentroids)
            self.iterations +=1
            self.labels=self.getLabels()
#             print(self.oldCentroids)
            self.getCentroids()
#             print(self.centroids)
#             print(self.oldCentroids)

    def initCentroids(self, init):
        if(len(init)==k):
            init=[i-1 for i in init]
            self.centroids=self.DB[init,:]
        elif(self.iterations==0):
            print("Incorrect initialization. Using random centroids.")
            perm=np.random.permutation(self.DB.shape[0])
            self.centroids=self.DB[perm[0:self.k]]

    def getCentroids(self):
        for i in np.unique(self.labelData[:,0]):
            self.labelCentroids[int(i)]=self.centroids[int(i)]=self.labelData[self.labelData[:
,0] == int(i)].mean(0)[1:]

    def getLabels(self):
        for i in range(len(self.DB)):
            dist = np.linalg.norm(self.DB[i] - self.centroids, ord=self.ord, axis=1)
            self.labelData[i][0] = np.argmin(dist)
        return self.labelData[:,0]

    def shouldStop(self):
        return ((np.linalg.norm(km.oldCentroids-km.centroids)==0)|(self.iterations>self.maxIte
rations))
```

```
# file=input()
file = sys.argv[1]
k = int(sys.argv[2])
try:
    init=eval(sys.argv[3])
except:
    init=[]
try:
    metricOrder=int(sys.argv[4])
except:
    metricOrder=2

data=np.genfromtxt(file, delimiter="\t")[:,2:]
X=(data - data.mean(0))
true_labels=np.array(list(pd.read_csv(file, sep='\t', lineterminator='\n', header=None).iloc[:
,1]))
km=KMEANS(k=k, metricOrder=metricOrder)
km.getDB(X)
km.initCentroids(init)

file = open("temp", "w")
file.write(str(km.k))
file.write("\n")
file.write(str(km.ord))
file.write("\n")
file.flush()
file.close()
np.save("centroids.npy",km.centroids)
np.savetxt("data.txt", km.DB)
```