```
# =============== Task 12 ===============

import math
import csv
import random

# <<<<< Compulsory Task 1 >>>>>
# K-Means clustering implementation

# <<<<< Compulsory Task 2 >>>>>
# In your main iteration loop, for each data point, calculate the squared distance
between
# each point and the cluster mean to which it belongs, and sum all of these squared
distances.
# Print out this sum once each iteration, and you can watch the objective function
converge.

# ====
# Define a function that computes the distance between two data points

def Distance(x1,y1,x2,y2):
      x = math.pow((x1-x2),2)
      y =   math.pow((y1-y2),2)
      d = math.sqrt(x+y)
      return d

# ====
# Define a function that reads data in from the csv files  HINT:
http://docs.python.org/2/library/csv.html

def Data_Input(file_name):
      data = []
      with open(file_name, 'rb') as csvfile:
            reader = csv.reader(csvfile)
            for row in reader:
                  data.append(row)
      return data

# ====
# Write the initialisation procedure

print "This is a programme to create clusters using K-Means >>>>>"
print ""
print "Choose one of these data sets: "
print "1) Data for 1953"
print "2) Data for 2008"
print "3) Data for Both"
File_Num = int(raw_input("Choose a number: "))

if File_Num == 1:
      File_Name = "data1953.csv"
elif File_Num == 2:
      File_Name = "data2008.csv"
elif File_Num == 3:
      File_Name = "dataBoth.csv"

Data = Data_Input(File_Name)

Country = [item[0] for item in Data]
```

```
Country_Names = Country[1:]

Birth_Rate = [item[1] for item in Data]
Birth_Rate_Nums = [float(element) for element in Birth_Rate[1:]]

Life_Exp = [item[2] for item in Data]
Life_Exp_Nums = [float(element) for element in Life_Exp[1:]]

print ""
Cluster_Num = int(raw_input("Choose the number of clusters to divide the data: "))

Int_Country = random.sample(Country_Names,Cluster_Num)

N = []
for i in range(0,Cluster_Num):
      N.append(Country_Names.index(Int_Country[i]))

B = []
for i in range(0,Cluster_Num):
      B.append(Birth_Rate_Nums[N[i]])

L = []
for i in range(0,Cluster_Num):
      L.append(Life_Exp_Nums[N[i]])

# ====
# Implement the k-means algorithm, using appropriate looping

Iter_Num = int(raw_input("Choose the number of iterations to run: "))
print ""

for i in range(0,Iter_Num):

      Cluster = {}

      for item in Country_Names:
            D = []
            index = Country_Names.index(item)
            for j in range(0,len(Int_Country)):

      D.append(Distance(Birth_Rate_Nums[index],Life_Exp_Nums[index],B[j],L[j]))
            D_Min = min(D)
            Cluster[item] = [D.index(D_Min), Birth_Rate_Nums[index],
Life_Exp_Nums[index], math.pow(D_Min,2)]

      B=[]
      L=[]
      C=[]
      D=[]

      for n in range(0,Cluster_Num):
            L_sum = 0
            B_sum = 0
            Count = 0
            D_sum = 0
            for v, k in Cluster.items():
                  if k[0] == n:
                        Count = 1 + Count
                        B_sum = k[1] + B_sum
                        L_sum = k[2] + L_sum
                        D_sum = k[3] + D_sum
            C.append(Count)
            B.append(B_sum/Count)
            L.append(L_sum/Count)
            D.append(D_sum)

      print "Iteration " + str(i+1) + " : " + str(D) # <<<<< Compulsory Task 2
>>>>>

# ====
# Print out the results

print ""
raw_input("Press 'Enter' to continue with the results >>>>>")

print ""
for i in range(0,len(C)):
      print "Number of countries in Cluster " + str(i + 1) + ": " + str(C[i])

print ""
for i in range(0,len(L)):
      print "The mean Life Expectancy in Cluster " + str(i + 1) + ": " + str(L[i])

print ""
for i in range(0,len(B)):
      print "The mean Birth Rate in Cluster " + str(i + 1) + ": " + str(B[i])

print ""
raw_input("Press 'Enter' to continue with the results >>>>>")

for n in range(0,Cluster_Num):
      print ""
      print "The Countries in Cluster " + str(n + 1) + ": "
      for v, k in Cluster.items():
            if k[0] == n:
                  print v
```