信息处理技术 作业5

From: 梁鑫宇 3160104494

题目：K-means聚类

> 尝试英文注释，尽量通过函数名等说明信息，减少不必要的注释

源代码

point.py

```python
class point:
    def __init__(self,value):
        self.value = value
        self.incluster = -1         # incluster represent which cluster this point belongs to

    def __lt__(self,other):         # overload '<' operator
        if isinstance(other,point):     # check the comparing object is a point
            return self.value < other.value
        else:
            return NotImplemented

    def __nearest(self,dists):      # find the minimum of the dists and return its position
        min = float('Inf')
        for dist in dists:
            if dist < min:
                min = dist
        return dists.index(min)

    def group(self,centroids):      # mark which cluster this point belongs to
        dists = []
        for centroid in centroids:  # calculate the distances between this point and every
centroid in centroids
            dists.append(abs(self.value-centroid))
        self.incluster = self.__nearest(dists)   # find the nearest centroid to this point and
assign the index to self.incluster
```

functions.py

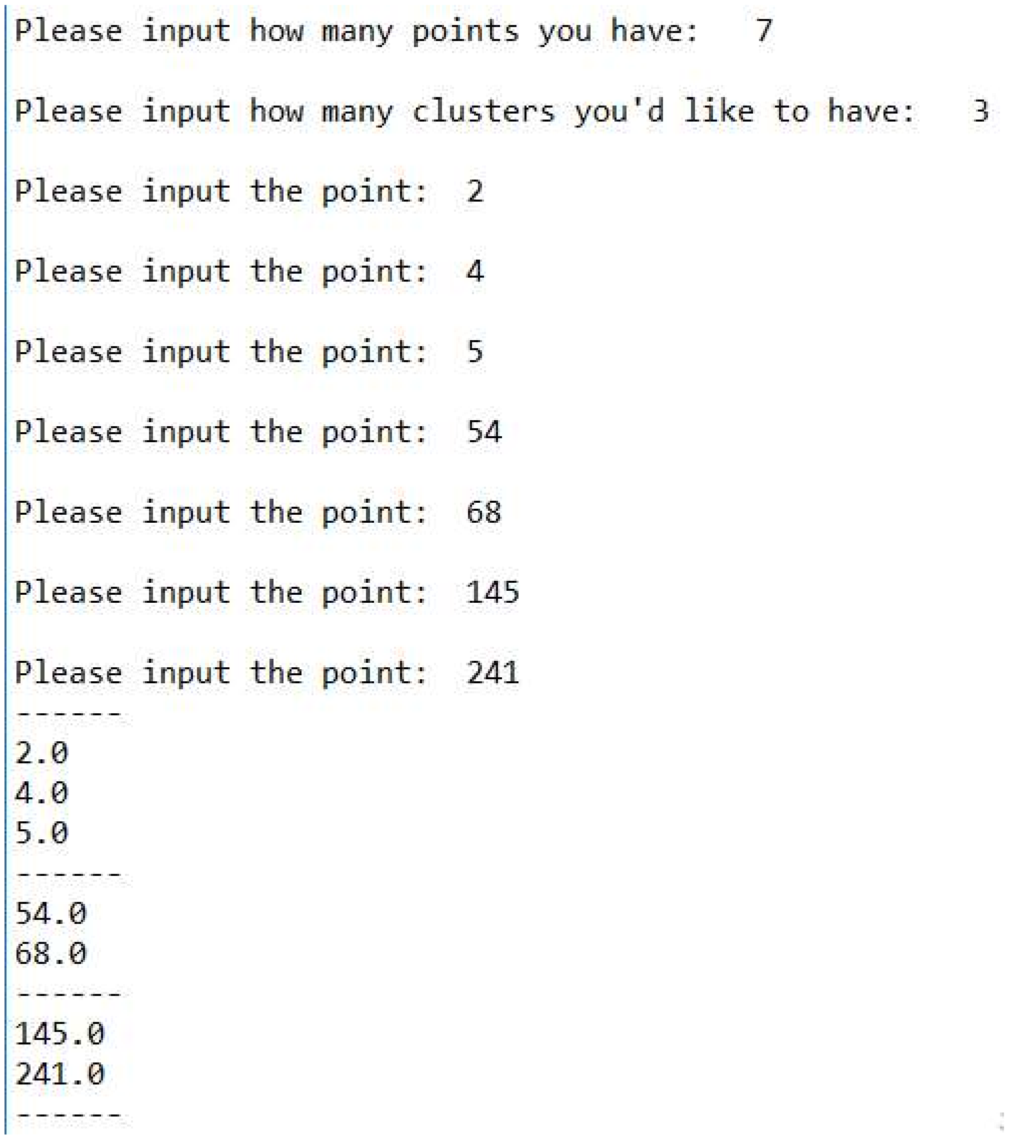
```python
from point import point
import random

def FreshClusters(points, Clusters):      #Clusters is a 2-dimension, k-length list, storing
point[]
    for Cluster in Clusters:
        Cluster.clear()
    for point in points:
        Clusters[point.incluster].append(point)

def avg(points):              # calculate the averange of a set of points
    sum = 0
    if len(points):
        for point in points:
            sum = sum + point.value
        return sum/len(points)
    else:
        print("Some centroids are too far from points")
        exit()

def SSE(Clusters,centroids):
    sse = 0
    for Cluster in Clusters:
        for point in Cluster:
            sse += pow(point.value-centroids[point.incluster],2)
    return sse

def TryNextLoop(points,centroids,Clusters):
    for Cluster in Clusters:
        centroids[Clusters.index(Cluster)] = avg(Cluster)
    for point in points:        # remark which cluster these points belong to
        point.group(centroids)
    FreshClusters(points, Clusters)
    return SSE(Clusters,centroids)


def adjust(points, centroids, Clusters):      #centroids is a list of centroid, with a size of k
    sse = SSE(Clusters,centroids)
    while 1:
        temp_points = points                 #back up
        temp_centroids = centroids
        temp_Clusters = Clusters
        new_sse = TryNextLoop(temp_points,temp_centroids,temp_Clusters)
        if new_sse < sse:
            points = temp_points             #fresh
            centroids = temp_centroids
            Clusters = temp_Clusters
            sse = new_sse
        else:
            break

def GenerateCentroids(points,K,Clusters):
    #---------      random       ----------
    # max = float("-Inf")
    # min = float("Inf")
    # for point in points:
    #     if(point.value>max):
    #         max = point.value
    #     if(point.value<min):
    #         min = point.value
    # Centroids = []
    # for i in range(0,K):
    #     Centroids.append(random.uniform(min,max))
    # for point in points:
    #     point.group(Centroids)
    # FreshClusters(points, Clusters)
    # return Centroids

    #--------      sort     ----------
    points.sort()
    avg_len = len(points)/K
    Centroids = []
    for i in range(0,K):
```

```python
        if i < K-1:
            Centroids.append(avg(points[int(i*avg_len):int((i+1)*avg_len)]))
        elif i == K-1:
            Centroids.append(avg(points[int(i * avg_len):]))
    for point in points:
        point.group(Centroids)
    FreshClusters(points, Clusters)
    return Centroids
```

main.py

```python
from point import point
import functions

N = input("Please input how many points you have:   ")
K = input("Please input how many clusters you'd like to have:   ")

points = []
Clusters = []

for i in range(0,int(K)):
    Clusters.append([])      # append empty lists as Clusters
for i in range(0,int(N)):
    value = input("Please input the point:  ")
    points.append(point(float(value)))  # create new point object with the input as the value and
append the objects into the points list

Centroids = functions.GenerateCentroids(points,int(K),Clusters)
functions.adjust(points,Centroids,Clusters)

print("------")
for Cluster in Clusters:
    for point in Cluster:
        print(point.value)
    print("------")
```

测试样例



```
Please input how many points you have:   7

Please input how many clusters you'd like to have:   3

Please input the point:  2

Please input the point:  4

Please input the point:  5

Please input the point:  54

Please input the point:  68

Please input the point:  145

Please input the point:  241
------
2.0
4.0
5.0
------
54.0
68.0
------
145.0
241.0
------
```