Q2

Import Libraries

```
import math
import numpy as np
import pandas as pd
from copy import deepcopy
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
from scipy.spatial import distance
%matplotlib inline
```

```
# number of datapoints
n = 10
data = np.array([[1,8], [1,1], [2,4], [3,3], [4,9], [4,6], [6,4], [7,7],
[9,9], [9,1]])
data_labels = np.array(['A','B','C','D','E','F','G','H','I','J'])
```

```
k = n
distances = np.zeros(shape=(n,n))
for i in range(n):
    distances[:,i] = np.linalg.norm(data - data[i], axis = 1)
print('Distances from-')
clusters = np.zeros(k)
clusters = deepcopy(data)
clusters_label = deepcopy(data_labels)
for cluster_label in clusters_label:
    print('\t{}'.format(cluster_label), end='')
print()
for i in range(k):
    print('{}'.format(clusters_label[i]), end='')
    for j in range(k):
        print ('\t{:.4f}'.format(distances[i][j]), end='')
    print()
```

```
Distances from-
    A   B   C   D   E   F   G   H   I   J
A   0.0000  7.0000  4.1231  5.3852  3.1623  3.6056  6.4031  6.0828  8.0623
10.6301
B   7.0000  0.0000  3.1623  2.8284  8.5440  5.8310  5.8310  8.4853  11.3137
8.0000
C   4.1231  3.1623  0.0000  1.4142  5.3852  2.8284  4.0000  5.8310  8.6023
7.6158
D   5.3852  2.8284  1.4142  0.0000  6.0828  3.1623  3.1623  5.6569  8.4853
6.3246
E   3.1623  8.5440  5.3852  6.0828  0.0000  3.0000  5.3852  3.6056  5.0000
9.4340
F   3.6056  5.8310  2.8284  3.1623  3.0000  0.0000  2.8284  3.1623  5.8310
7.0711
G   6.4031  5.8310  4.0000  3.1623  5.3852  2.8284  0.0000  3.1623  5.8310
4.2426
H   6.0828  8.4853  5.8310  5.6569  3.6056  3.1623  3.1623  0.0000  2.8284
6.3246
I   8.0623  11.3137 8.6023  8.4853  5.0000  5.8310  5.8310  2.8284  0.0000
8.0000
J   10.6301 8.0000  7.6158  6.3246  9.4340  7.0711  4.2426  6.3246  8.0000
0.0000
```

```python
# Single Link
# Cluster 0 {B,C,D,E,F,G,H,I}
# Cluster 1 {J}
# Cluster 2 {A}
k = 3
single_link_centroids = np.zeros(shape=(k,2))
clusters = np.zeros(10)
clusters = [2,0,0,0,0,0,0,0,0,1]

# Calculating new centroids
# new_centroids[i] = np.mean(data[clusters == i], axis=0)
for i in range(k):
    # print('Cluster points: {}'.format([list(x) for x,y in zip(data,clusters)
if y == i]))
    single_link_centroids[i] = np.mean([list(x) for x,y in zip(data,clusters)
if y == i], axis=0)

ss = 0
for i in range(k):
        cluster_points = [list(x) for x,y in zip(data,clusters) if y == i]
```

```
        # print('Cluster Points: {}'.format(cluster_points))
        distances = np.linalg.norm(cluster_points - single_link_centroids[i],
axis = 1)
        ss += sum([x*x for x in distances ])
print('Single Link SSE: {}'.format(ss))
```

```
Single Link SSE: 107.875
```

```
# Complete Link
# Cluster 0 {A,B,C,D,F,G,E}
# Cluster 1 {J}
# Cluster 2 {H,I}
k = 3
single_link_centroids = np.zeros(shape=(k,2))
clusters = np.zeros(10)
clusters = [0,0,0,0,0,0,0,2,2,1]

# Calculating new centroids
# new_centroids[i] = np.mean(data[clusters == i], axis=0)
for i in range(k):
    # print('Cluster points: {}'.format([list(x) for x,y in zip(data,clusters)
if y == i]))
    single_link_centroids[i] = np.mean([list(x) for x,y in zip(data,clusters)
if y == i], axis=0)

ss = 0
for i in range(k):
        cluster_points = [list(x) for x,y in zip(data,clusters) if y == i]
        # print('Cluster Points: {}'.format(cluster_points))
        distances = np.linalg.norm(cluster_points - single_link_centroids[i],
axis = 1)
        ss += sum([x*x for x in distances ])
print('Complete Link SSE: {}'.format(ss))
```

```
Complete Link SSE: 72.0
```