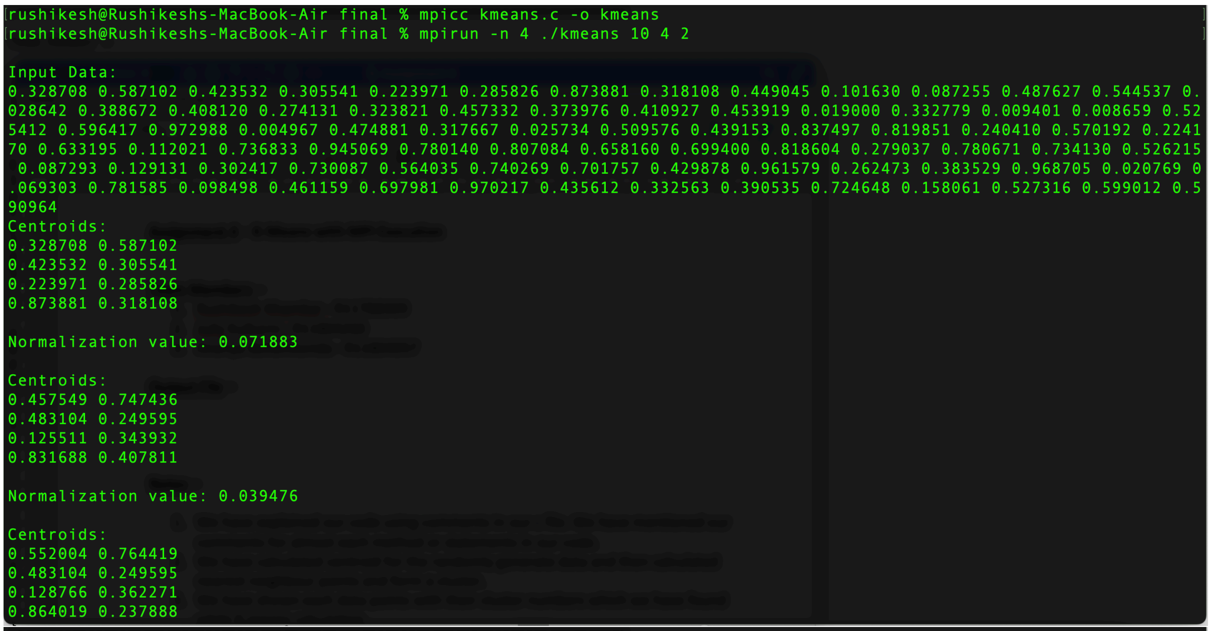
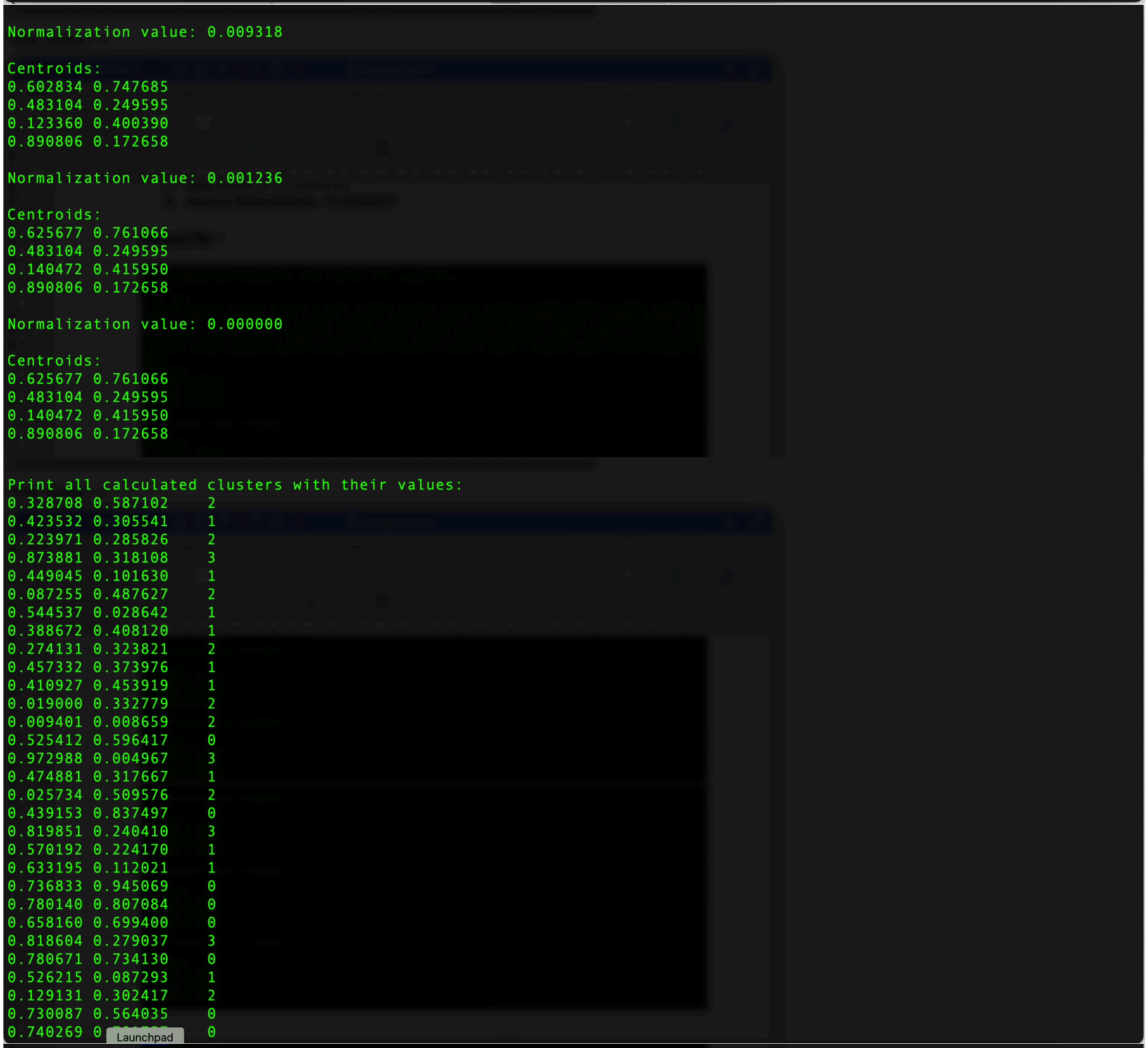
Assignment 4 – K-Means with MPI Execution

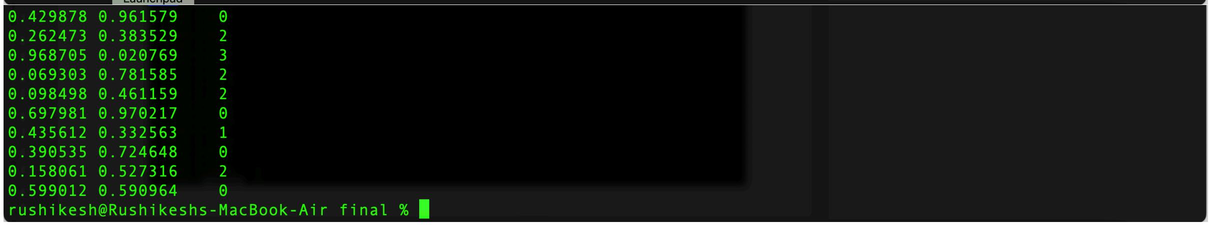
Output File –



```
rushikesh@Rushikeshs-MacBook-Air final % mpicc kmeans.c -o kmeans
rushikesh@Rushikeshs-MacBook-Air final % mpirun -n 4 ./kmeans 10 4 2

Input Data:
0.328708 0.587102 0.423532 0.305541 0.223971 0.285826 0.873881 0.318108 0.449045 0.101630 0.087255 0.487627 0.544537 0.
028642 0.388672 0.408120 0.274131 0.323821 0.457332 0.373976 0.410927 0.453919 0.019000 0.332779 0.009401 0.008659 0.52
5412 0.596417 0.972988 0.004967 0.474881 0.317667 0.025734 0.509576 0.439153 0.837497 0.819851 0.240410 0.570192 0.2241
70 0.633195 0.112021 0.736833 0.945069 0.780140 0.807084 0.658160 0.699400 0.818604 0.279037 0.780671 0.734130 0.526215
 0.087293 0.129131 0.302417 0.730087 0.564035 0.740269 0.701757 0.429878 0.961579 0.262473 0.383529 0.968705 0.020769 0
.069303 0.781585 0.098498 0.461159 0.697981 0.970217 0.435612 0.332563 0.390535 0.724648 0.158061 0.527316 0.599012 0.5
90964
Centroids:
0.328708 0.587102
0.423532 0.305541
0.223971 0.285826
0.873881 0.318108

Normalization value: 0.071883

Centroids:
0.457549 0.747436
0.483104 0.249595
0.125511 0.343932
0.831688 0.407811

Normalization value: 0.039476

Centroids:
0.552004 0.764419
0.483104 0.249595
0.128766 0.362271
0.864019 0.237888
```



```
Normalization value: 0.009318

Centroids:
0.602834 0.747685
0.483104 0.249595
0.123360 0.400390
0.890806 0.172658

Normalization value: 0.001236

Centroids:
0.625677 0.761066
0.483104 0.249595
0.140472 0.415950
0.890806 0.172658

Normalization value: 0.000000

Centroids:
0.625677 0.761066
0.483104 0.249595
0.140472 0.415950
0.890806 0.172658


Print all calculated clusters with their values:
0.328708 0.587102    2
0.423532 0.305541    1
0.223971 0.285826    2
0.873881 0.318108    3
0.449045 0.101630    1
0.087255 0.487627    2
0.544537 0.028642    1
0.388672 0.408120    1
0.274131 0.323821    2
0.457332 0.373976    1
0.410927 0.453919    1
0.019000 0.332779    2
0.009401 0.008659    2
0.525412 0.596417    0
0.972988 0.004967    3
0.474881 0.317667    1
0.025734 0.509576    2
0.439153 0.837497    0
0.819851 0.240410    3
0.570192 0.224170    1
0.633195 0.112021    1
0.736833 0.945069    0
0.780140 0.807084    0
0.658160 0.699400    0
0.818604 0.279037    3
0.780671 0.734130    0
0.526215 0.087293    1
0.129131 0.302417    2
0.730087 0.564035    0
0.740269 0           0
```



```
0.429878 0.961579    0
0.262473 0.383529    2
0.968705 0.020769    3
0.069303 0.781585    2
0.098498 0.461159    2
0.697981 0.970217    0
0.435612 0.332563    1
0.390535 0.724648    0
0.158061 0.527316    2
0.599012 0.590964    0
rushikesh@Rushikeshs-MacBook-Air final % 
```

<u>Notes</u> –

1. We have explained our code using comments in our c file. We have mentioned our comments for almost each method or statements in our code.
2. We have calculated centroid for the randomly generate data and then calculated nearest neighbour points and form a cluster.
3. We have shown each data points with their cluster numbers which we have found while K-means calculation.
4. For Screenshot purpose, we just have shown 2 dimensional data with size 10 and consider 4 K-clusters only. But, we can calculate data for 16-dimensional data with 10000 data size.

<u>Execution in the code -</u>

1. Processor 0 generates random data using create_datapoints function and share it with other processor using MPI_Scatter function.
2. Each other individual processor calculate centroid using normalisation and broadcast with other processor using MPI_BCast to maintain consistency among all the processors.
3. After calculation of the centroid, all the processors gathers all the data using MPI_Reduce
4. After getting centroid, each processor will segregate the data with respected clusters.
5. Processor 0 will just help us to print all the calculated clusters at the end of the execution.

<u>Steps -</u>

1. For C Code compilation - mpicc kmeans.c -o kmeans
2. For execution, you need to hit below command line in terminal –
   mpirun -n 4 ./kmeans 10 4 2

   mpirun -n 4 – indicates how many processors, we are going to use for execution
   ./kmeans 10 4 2 – indicates 10 (total number of data points), 4 (K- clusters for computation) and 2 (n-dimensional data)

Thank you for such thoughtful assignment.