HW3 – Wenyue Wang

- My program includes with three files, a main.c that declare matrices and vectors as well as implement the whole while loop for the training. A kmeans.c file that include all helper functions and a kmeans.h that include all header files.
- In main.c, the whole procedure is:
    1) Let n = data_num, d = dimension_num, c = cluster_num
    2) Read in .txt file and saved in a data_matrix (d * n), also we can apply normalization
    3) Use a (d * c) matrix to store initial defined centroids, can be either first c points or randomly select
    4) Use a (c * n) matrix to store distance, from each data point to each centroid
    5) Use a vector with length d to store which cluster the point belongs to
    6) Use a vector with length c to store how many points in each cluster
    7) Calculate the root mean square
    8) Create a while loop to :
        1) Update centroids
        2) Repeat steps 4, 5, 6, 7
        3) Calculate the root mean square difference after each iteration
    9) Stop the while loop when either the root mean square difference less that threshold or reach max iteration number
    10) Free up all created matrices and vectors

- Test read in invalid dataset, it will show cannot read in data

```
summerwang@lawn-128-61-56-202 HW3 % ./output.out invalid.txt 3
 ---------------------- Initialize --------------------
ERROR: Cannot open file for reading in data
```

- Test read in data with different dimensions:
  - test with both WineData_2col.txt and WineData_3col.txt, and it works well
- Test with different clusters:
  - I created a short dataset with 12 points,
  - and I test it with cluster number 0, 3, 12, 13.

```
summerwang@lawn-128-61-56-202 HW3 % ./output.out test.txt 0
The k-value input was:  0
K must be larger than zero, and no larger than 2147483647.
```

```
summerwang@lawn-128-61-56-202 HW3 % ./output.out test.txt 12
 ---------------------- Initialize --------------------
cluster 0:  3
cluster 1:  6
cluster 2:  3
cluster 3:  0
cluster 4:  0
cluster 5:  0
cluster 6:  0
cluster 7:  0
cluster 8:  0
cluster 9:  0
cluster 10:  0
cluster 11:  0
RMS: 0.00
```

```
summerwang@lawn-128-61-56-202 HW3 % ./output.out test.txt 3
 ---------------------- Initialize --------------------
cluster 0:  3
cluster 1:  9
cluster 2:  0
RMS: 288.00
```

```
summerwang@lawn-128-61-56-202 HW3 % ./output.out test.txt 13
 ---------------------- Initialize --------------------
ERROR: cannot initialize centroids
ERROR: cannot initialize centroids
ERROR: cannot get distance matrix
ERROR: cannot create cluster vector
ERROR: cannot get how many points in each cluster
ERROR: cannot calculate root mean square
```

- For normalization, I tested with both 2col data and 3col data, and choose to randomly selected the initial centroids. Also, I set the max iteration numbers be 100, and threshold be 0.001.

	2col – 3 clusters	**2col – 6 clusters**	**3col – 3 clusters**	**3col – 6 clusters**
Origin data – t1	5 iters	23 iters	6 iters	26 iters
Origin data – t2	4 iters	12 iters	9 iters	20 iters
Origin data – t3	7 iters	19 iters	10 iters	17 iters
Normalized – t1	2 iters	1 iter	2 iters	2 iters
Normalized – t2	2 iters	2 iters	2 iters	2 iters
Normalized – t3	2 iters	2 iters	2 iters	2 iters

From the results we can see by normalizing data first, we can obviously reduce the iteration times to reach the desired training result.

- To test random, I used own original dataset and not normalized one.

```
12 2
0       1
1       0
60      60
1       1
70      67
100     2
100     1
70      70
101     0
110     1
80      75
105     2
```

	K = 3
First k	Initial: (2, 1, 9) Final: (2, 1, 9) 2 iters
Random – t1	Initial: (2, 5, 5) Final: (5, 3, 4) 3 iters
Random – t2	Initial: (4, 3, 5) Final: (4, 3, 5) 2 iters
Random – t3	Initial: (7, 2, 3) Final: (7, 3, 2) 3 iters

Based on the test dataset, the whole set should be divided into three clusters:

1. (0,1), (1,0), (1,1)
2. (60,60), (70,67), (70, 70), (80, 75)
3. (100, 2), (100, 1), (101, 0), (110, 1), (105,2)

Based on the results, we can see if we just use first three points as initial centroids, the training will in local optimize. But if we randomly choose initial centroids, only the third trial we did not get the correct result. So randomly choose initial centroids is more desirable.

- For WineData_2col, I think k = 2 would be best for classification. I tested with normalized data and randomly select initial centroids. By testing k from 2 to 6, I found that mostly only 2 clusters are useful, like with actual numbers in it. In addition, I printed out the normalized value, and all normalized data are either ~39.987 or ~-39.987
- For WineData_3col, I think k = 3 would be best. I did the similar way by testing from k = 2 to 6, and most results have 3 clusters with numbers.
- In conclusion, I believe this method produces meaning results for both dataset.