a6/kmeans/file_io.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>     /* strtok() */
#include <sys/types.h>  /* open() */
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>     /* read(), close() */
#include <mpi.h>

#include "kmeans.h"

double * dataset_generation(int numObjs, int numCoords, long *rank_numObjs)
{
    double * objects = NULL, * rank_objects = NULL;
    long i, j, k;

    // Random values that will be generated will be between 0 and 10.
    double val_range = 10;

    int rank, size;
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    /*
     * TODO: Calculate number of objects that each rank will examine (*rank_numObjs)
     */
    *rank_numObjs = numObjs / size;
    if (rank < numObjs % size) {
        (*rank_numObjs)++;
    }


    /* allocate space for objects[][] and read all objects */
    int sendcounts[size], displs[size];
    if (rank == 0) {
        objects = (typeof(objects)) malloc(numObjs * numCoords * sizeof(*objects));

        /*
         * TODO: Calculate sendcounts and displs, which will be used to scatter data to each rank.
         * Hint: sendcounts: number of elements sent to each rank
         *       displs: displacement of each rank's data
         */

        int count = 0;
        int remainder = numObjs % size;

        for (k = 0; k < size; k++) {

            int k_numObjs = numObjs / size;
            if (k < remainder) {
                k_numObjs++;
```

```
52              }
53
54              sendcounts[k] = k_numObjs * numCoords;
55              displs[k] = count;
56              count += sendcounts[k];
57          }
58      }
59
60      /*
61       * TODO: Broadcast the sendcounts and displs arrays to other ranks
62       */
63      MPI_Bcast(sendcounts, size, MPI_INT, 0, MPI_COMM_WORLD);
64      MPI_Bcast(displs, size, MPI_INT, 0, MPI_COMM_WORLD);
65
66
67      /* allocate space for objects[][] (for each rank separately) and read all objects */
68      rank_objects = (typeof(rank_objects)) malloc((*rank_numObjs) * numCoords *
    sizeof(*rank_objects));
69
70      /* rank 0 will generate data for the objects array. This array will be used later to
    scatter data to each rank. */
71      if (rank == 0) {
72          for (i=0; i<numObjs; i++)
73          {
74              unsigned int seed = i;
75              for (j=0; j<numCoords; j++)
76              {
77                  objects[i*numCoords + j] = (rand_r(&seed) / ((double) RAND_MAX)) *
    val_range;
78                  if (_debug && i == 0)
79                      printf("object[i=%ld][j=%ld]=%f\n",i,j,objects[i*numCoords + j]);
80              }
81          }
82      }
83
84      /*
85       * TODO: Scatter objects to every rank. (hint: each rank may receive different number
    of objects)
86       */
87      MPI_Scatterv(objects, sendcounts, displs, MPI_DOUBLE, rank_objects, sendcounts[rank],
    MPI_DOUBLE, 0, MPI_COMM_WORLD);
88
89
90      if (rank == 0)
91          free(objects);
92
93      return rank_objects;
94  }
95
```