#include<mpi.h>

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

int main (int argc, char \*\*argv) {

 int rank,size, length, data, \*buffer;

 char filename[64]=“data.txt”;

 MPI\_File fileHandle;

 MPI\_Status status;

MPI\_Offset fileSize;

 MPI\_Init(&argc, &argv);

 MPI\_Comm\_rank (MPI\_COMM\_WORLD, &rank);

 MPI\_Comm\_size (MPI\_COMM\_WORLD, &size);

 If (rank==0) {

 MPI\_File\_open(MPI\_COMM\_SELF, filename, MPI\_MODE\_RDONLY, MPI\_INFO\_NULL, &fileHandle);

 MPI\_File\_get\_size(fileHandle, &fileSize);

 MPI\_File\_close(&fileHandle);

 length=(int) (fileSize/size);

 data=(int) (length/sizeof(int));

 }

 MPI\_Bcast(&fileSize,1,MPI\_INT,0,MPI\_COMM\_WORLD);

 MPI\_Bcast(&length,1,MPI\_INT,0,MPI\_COMM\_WORLD);

 MPI\_Bcast(&data,1,MPI\_INT,0,MPI\_COMM\_WORLD);

 buffer=(int\*) malloc(data\*sizeof(int));

 if(!buffer) {

 printf(“No memory available\n”);

 MPI\_finalize();

 return(-1);

 }

 MPI\_File\_open(MPI\_COMM\_WORLD, filename, MPI\_MODE\_RDWR, MPI\_INFO\_NULL, &fileHandle);

 MPI\_File\_seek(fileHandle, rank\*length, MPI\_SEEK\_SET);

 MPI\_File\_read(fileHandle, buffer, data, MPI\_INT, &status);

 MPI\_File\_close(&fileHandle);

 for(int i=0; i<5; i++)

 printf(“Process %d --> buffer[%d]=%02d\n”, rank,I,buffer[i]);

 printf(“\n”);

 free(buffer);

 MPI\_Finalize();

 return(0);

}