#include<mpi.h> #include<stdio.h> #include<limits.h> #include<math.h> #include<string.h> #include<stdlib.h> #define MAXSIZE 100

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

{

float startTime, endTime, execTime; long int s, s0, startIndex, endIndex; float data[MAXSIZE];

float local\_min = INT\_MAX, global\_min, global\_max, local\_max = INT\_MIN; long int i;

int namelen; MPI\_Init(&argc, &argv);

int myid;//myid is rank of processor MPI\_Comm\_rank(MPI\_COMM\_WORLD, &myid); int numprocs;//number of processors MPI\_Comm\_size(MPI\_COMM\_WORLD, &numprocs);

char processor\_name[MPI\_MAX\_PROCESSOR\_NAME];//processor name MPI\_Get\_processor\_name(processor\_name, &namelen); fprintf(stderr,"process %d is on %s \n", myid, processor\_name); fflush(stderr);

//data initialization for(i=0;i<MAXSIZE;i++)

data[i] =i+1; if(myid==0)

{

s=(int)(MAXSIZE/numprocs); s0= s+ (MAXSIZE%numprocs);

printf("s=%ld, s0=%ld \n", s,s0);

}

else

{

s=(int) (MAXSIZE/numprocs); s0= s+ (MAXSIZE%numprocs); startIndex = s0+(myid-1) \*s; endIndex = startIndex +s;

}

MPI\_Bcast(&s, 1, MPI\_FLOAT, 0,MPI\_COMM\_WORLD); MPI\_Bcast(&s0, 1, MPI\_FLOAT, 0,MPI\_COMM\_WORLD);

execTime=0; if(myid==0)

{

startTime=MPI\_Wtime();

}

if(myid==0)

{

for(i=0; i<s0;i++)

{

if(local\_max<data[i]){

local\_max = data[i];

}

if(local\_min>data[i]){ local\_min = data[i];

}

}

printf("Local maximum from master = %f from processor %d \n", local\_max,myid);

printf("Local minimum from master = %f from processor %d \n", local\_min,myid);

}

else

{

//worker local maxima and local minima local\_max = INT\_MIN;

local\_min = INT\_MAX; for(i=startIndex; i<endIndex;i++)

{

if(local\_max<data[i]){ local\_max = data[i];

}

if(local\_min>data[i]){ local\_min = data[i];

}

}

printf("Local maximum from worker = %f from processor %d \n", local\_max,myid);

printf("Local maximum from worker = %f from processor %d \n", local\_min,myid);

}

//Global Max AND Global Min MPI\_Reduce(&local\_max,&global\_max,1, MPI\_FLOAT, MPI\_MAX,0, MPI\_COMM\_WORLD);

MPI\_Reduce(&local\_min,&global\_min,1, MPI\_FLOAT, MPI\_MIN,0, MPI\_COMM\_WORLD);

if(myid==0)

{

endTime = MPI\_Wtime(); execTime= endTime-startTime;

printf("Global Maximum is %f \n",global\_max); printf("Global Minimum is %f \n",global\_min); printf("Execution Time is %f sec \n ",execTime);

}

MPI\_Finalize();

}