PI VALUE USING MONTE CARLO METHOD

**SOURCE CODE**

#include <stdio.h> #include <stdlib.h> #include "mpi.h" #include <math.h> #include<time.h> #define SEED 35791246

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

{

long niter = 1000000;

int myid; *//holds process's rank id*

double x,y; *//x,y value for the rand pts*

int i, count=0;

double z; *//Used to check if x^2+y^2<=1*

double pi; *//holds approx value of pi*

int nodenum; double st = clock();

MPI\_Init(&argc, &argv); *//Start MPI* MPI\_Comm\_rank(MPI\_COMM\_WORLD, &myid); *//get rank of node's* process MPI\_Comm\_size(MPI\_COMM\_WORLD, &nodenum);

int recieved[nodenum]; long recvniter[nodenum];

srand(SEED+myid); *//Give rand() a seed value*

if(myid != 0)

{

for (i=0; i<niter; ++i) *//main loop*

{

x= ((double)rand())/RAND\_MAX; *//gets a random x* coordinate y =((double)rand())/RAND\_MAX; *//gets a random y* coordinate z = sqrt(x\*x+y\*y); *//Checks if* inside unit circle

if (z<=1)

{

count++;

}

}

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

{

MPI\_Send(&count,

1, MPI\_INT, 0,

1, MPI\_COMM\_WORLD);

MPI\_Send(&niter,

1, MPI\_LONG, 0,

2, MPI\_COMM\_WORLD);

}

}

else if (myid == 0)

{

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

{

MPI\_Recv(&recieved[i],

nodenum, MPI\_INT, MPI\_ANY\_SOURCE, 1,

MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

MPI\_Recv(&recvniter[i],

nodenum, MPI\_LONG, MPI\_ANY\_SOURCE, 2,

MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

}

}

if (myid == 0) *//if root process*

{

int finalcount = 0; long finalniter = 0;

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

{

finalcount += recieved[i]; finalniter += recvniter[i];

}

pi = ((double)finalcount/(double)finalniter)\*4.0; printf("Pi: %f", pi); *//Print the calculated value of pi*

double en = clock();

printf("\nTime: %lf",(en-st)/CLOCKS\_PER\_SEC);

}

MPI\_Finalize(); *//Close the MPI instance*

return 0;

}