/\* MPI program that uses a monte carlo method to compute the value of PI \*/

#include &lt;stdlib.h&gt;

#include &lt;stdio.h&gt;

#include &lt;math.h&gt;

#include &lt;string.h&gt;

#include &lt;stdio.h&gt;

#include &lt;sprng.h&gt;

#include &lt;mpi.h&gt;

#define USE\_MPI

#define SEED 35791246

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

{

int niter=0;

double x,y;

int i,j,count=0,mycount; /\* # of points in the 1st quadrant of unit

circle \*/

double z;

double pi;

int myid,numprocs,proc;

MPI\_Status status;

int master =0;

int tag = 123;

int \*stream\_id; /\* stream id generated by SPRNGS \*/

MPI\_Init(&amp;argc,&amp;argv);

MPI\_Comm\_size(MPI\_COMM\_WORLD,&amp;numprocs);

MPI\_Comm\_rank(MPI\_COMM\_WORLD,&amp;myid);

if (argc &lt;=1) {

fprintf(stderr,&quot;Usage: monte\_pi\_mpi number\_of\_iterations\n&quot;);

MPI\_Finalize();

exit(-1);

}

sscanf(argv[1],&quot;%d&quot;,&amp;niter); /\* 1st argument is the number of

iterations\*/

/\* initialize random numbers \*/

stream\_id = init\_sprng(myid,numprocs,SEED,SPRNG\_DEFAULT);

mycount=0;

for ( i=0; i&lt;niter; i++) {

x = (double)sprng(stream\_id);

y = (double)sprng(stream\_id);

z = x\*x+y\*y;

if (z&lt;=1) mycount++;

}

if (myid ==0) { /\* if I am the master process gather results from others

\*/

count = mycount;

for (proc=1; proc&lt;numprocs; proc++) {

MPI\_Recv(&amp;mycount,1,MPI\_REAL,proc,tag,MPI\_COMM\_WORLD,&amp;status);

count +=mycount;

}

pi=(double)count/(niter\*numprocs)\*4;

printf(&quot;\n # of trials= %d , estimate of pi is %g

\n&quot;,niter\*numprocs,pi);

}

else { /\* for all the slave processes send results to the master \*/

printf(&quot;Processor %d sending results= %d to master

process\n&quot;,myid,mycount

);

MPI\_Send(&amp;mycount,1,MPI\_REAL,master,tag,MPI\_COMM\_WORLD);

}

MPI\_Finalize(); /\* let MPI finish up \*/

}