

22. Write a program to determine the total of a large array using all processes available on a cluster. Assume a large array and a cluster of around 10 processes. Hint: Use loop splitting discussed in class where each process computes the partial sums of some of the elements.

Parallel Execution

```
#include <stdio.h>
#include <iostream>
#include <mpi.h>
using namespace std;
int main(int argc, char** argv)
{
    int size, node, i, sum=0, n1=0;
    MPI_Status status;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    int a[] = { 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30};
    int g = sizeof(a)/sizeof(a[1]);
    for(i=node; i < g; i=i+size)
    {
        sum=sum+a[i];
    }
    MPI_Reduce(&sum, &n1, 1, MPI_INT, MPI_SUM, 0, MPI_COMM_WORLD);
    if(node==0)
        cout<<"Total sum at node"<<node<<" "<<n1<<endl;
    MPI_Finalize();
    return 0;
}
```

23. Write the sequential Rank Sort algorithm. Write the necessary outline to implement it on oscar.calstatela.edu assuming the array size is equal to the cluster size. i.e., each process determines the rank of an element that is sent to the head node to compile the sorted array.

Sequential

```
#include <stdio.h>
#include <iostream>
#include <mpi.h>
using namespace std;
int main(int argc, char** argv)
{
    int size, node, i, sum=0, n1=0;
    MPI_Status status;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    int a[] = { 23,32,0,2,45,67,90,76,212};
    int g = sizeof(a)/sizeof(a[1]);
```

```

int b[g];
int rank=0;
for(int x=0;x<g;x++)
{
    rank=0;
    for(int y=0;y<g;y++)
        if(a[x]>a[y])
            rank++;
    b[rank]=a[x];
}
for(i=0; i< g; i=i++)
{
    cout<<"Node has Sum "<<b[i]<<endl;
}
MPI_Finalize();
return 0;
}

```

Parallel:

```

#include <stdio.h>
#include <iostream>
#include <mpi.h>
using namespace std;
int main(int argc, char** argv)
{
    int size, node,sum=0, n1=0;
    MPI_Status status;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    int a[]= {23,32,0,2,45,67,90,76,212};
    int g= sizeof(a)/sizeof(a[1]);
    int rank=0;
    for (int i = 0; i < g; i++)
    {
        if(a[node]>a[i])
            rank++;
    }
    if(node!=0)
        MPI_Send(&rank,1,MPI_INT,0,0,MPI_COMM_WORLD);
    else{
        int b[9];
        b[rank]=a[node];
        for(int x=1;x<g;x++)
        {
            MPI_Recv(&rank,1,MPI_INT,x,0,MPI_COMM_WORLD,&status);
            b[rank]=a[x];
        }
        cout<<"Sorted Array ";
        for (int r = 0; r < g; r++)

```

```

        cout<<b[r]<<" ";
        cout<<endl;

    }
    MPI_Finalize();
    return 0;

}

```

24. Compute $\sum 1 / (1 + x^2)$ starting with $x=0$ and intervals of 0.02 until $x=1$ (use loop splitting with n processes).

Parallel:

```

#include <stdio.h>
#include <iostream>
#include <mpi.h>
using namespace std;
int main(int argc, char** argv)
{
    int size, node;
    MPI_Status status;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    double sum=0, v=0, r, sum1=0;

    for (int i = 0; i < 1/0.02; i=i+size)
    {
        r=0.02*i;
        sum1=(1/(1+(r*r)));
        v=v+sum1;
        cout<<"Value of X: "<<r<<" "<<sum1<<endl;
    }
    cout<<"Node "<<node<<" has Sum"<<v<<endl;
    MPI_Reduce(&v,&sum,1,MPI_Double,MPI_Sum,0,MPI_COMM_WORLD);
    if(node==0)
        cout<<"Total Sum = "<<sum<<endl;
    MPI_Finalize();
    return 0;
}

```

```
cs5440s23@oscar:~  
Total Sum = 41.6866  
[cs5440s23@oscar ~]$ mpicc -o p242 242.cpp  
242.cpp:27:3: warning: no newline at end of file  
[cs5440s23@oscar ~]$ mpirun N p242  
Value of X: 0 1  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Node 1 has Sum4.16866  
Value of X: 0 1  
Value of X: 0 1  
Value of X: 0.2 0.961538  
Value of X: 0 1  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0 1  
Value of X: 0.4 0.862069  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Value of X: 0 1  
Value of X: 0 1  
Value of X: 0 1  
Value of X: 0 1  
Value of X: 0.4 0.862069  
Node 2 has Sum4.16866  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Node 6 has Sum4.16866  
Node 7 has Sum4.16866  
Value of X: 0.2 0.961538  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Node 6 has Sum4.16866  
Node 7 has Sum4.16866  
Value of X: 0.2 0.961538  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Node 5 has Sum4.16866  
Node 3 has Sum4.16866  
Value of X: 0.8 0.609756  
Value of X: 0.6 0.735294  
Value of X: 0.6 0.735294  
Node 4 has Sum4.16866  
Value of X: 0.8 0.609756  
Node 8 has Sum4.16866  
Value of X: 0.8 0.609756  
Node 9 has Sum4.16866  
Value of X: 0 1  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Node 0 has Sum4.16866  
Total Sum = 41.6866  
[cs5440s23@oscar ~]$
```

```
cs5440s23@oscar:~  
Value of X: 0.8 0.609756  
Value of X: 0 1  
Value of X: 0 1  
Value of X: 0 1  
Value of X: 0 1  
Value of X: 0.4 0.862069  
Node 2 has Sum4.16866  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Node 6 has Sum4.16866  
Node 7 has Sum4.16866  
Value of X: 0.2 0.961538  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Node 5 has Sum4.16866  
Node 3 has Sum4.16866  
Value of X: 0.8 0.609756  
Value of X: 0.6 0.735294  
Value of X: 0.6 0.735294  
Node 4 has Sum4.16866  
Value of X: 0.8 0.609756  
Node 8 has Sum4.16866  
Value of X: 0.8 0.609756  
Node 9 has Sum4.16866  
Value of X: 0 1  
Value of X: 0.2 0.961538  
Value of X: 0.4 0.862069  
Value of X: 0.6 0.735294  
Value of X: 0.8 0.609756  
Node 0 has Sum4.16866  
Total Sum = 41.6866  
[cs5440s23@oscar ~]$
```

25 : e x is evaluated as a series given below. The result is more accurate for a large number (say 100) of terms. Write the factorial as a real number (rather than an integer as it will result in an overflow error).

Sequential

```

#include <stdio.h>
#include <iostream>
#include <mpi.h>
#include <math.h>
using namespace std;
double fact1(double x){
    double fact=1;
    while(x>=1){
        fact=fact*x;
        x--;
    }
    return fact;
}
int main(int argc, char** argv)
{
    int size, node;
    MPI_Status status;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    double v=0, r=2,sum=0,sum1=0;
    for (double i=0;i<=100;i++)
    {
        sum1= pow(r,i);
        sum=sum1/fact1(i);
        v=v+sum;
        cout<<" "<<i<<"has Value"<<sum<<" "<<endl;
    }
    cout<<"Total Value is " <<v<<endl;
    MPI_Finalize();
    return 0;
}

```

Parallel

```

#include <stdio.h>
#include <iostream>
#include <mpi.h>
#include <math.h>
using namespace std;
double fact1(double x){
    double fact=1;
    while(x>=1){
        fact=fact*x;
        x--;
    }
    return fact;
}
int main(int argc, char** argv)
{
    int size, node;
    MPI_Status status;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);

```

```

MPI_Comm_size(MPI_COMM_WORLD, &size);
double v=0, r=2,sum=0,sum1=0, t;
for (double i=0;i<=100;i=i+size)
{
    sum1= pow(r,i);
    sum=sum1/fact1(i);
    v=v+sum;
    cout<<" "<<i<<"has Value"<<sum<<" "<<endl;

}
cout<<"Node has " <<node<<"value " <<v<<endl;
MPI_Reduce(&v,&t,1,MPI_DOUBLE,MPI_SUM,0,MPI_COMM_WORLD);
if(node==0)
    cout<<"Total Value is " <<t<<endl;
MPI_Finalize();
return 0;
}

```

```

cs5440s23@oscar:~
60has Value1.38556e-64
70has Value9.85586e-80
70has Value9.85586e-80
60has Value1.38556e-64
80has Value1.68916e-95
80has Value1.68916e-95
60has Value1.38556e-64
70has Value9.85586e-80
Node has 6value 1.00028
60has Value1.38556e-64
80has Value1.68916e-95
70has Value9.85586e-80
70has Value9.85586e-80
80has Value1.68916e-95
80has Value1.68916e-95
90has Value8.33228e-112
90has Value8.33228e-112
100has Value1.3583e-128
100has Value1.3583e-128
90has Value8.33228e-112
Node has 3value 1.00028
Node has 5value 1.00028
100has Value1.3583e-128
90has Value8.33228e-112
90has Value8.33228e-112
Node has 4value 1.00028
100has Value1.3583e-128
100has Value1.3583e-128
Node has 8value 1.00028
Node has 9value 1.00028
0has Value1
10has Value0.000282187
20has Value4.30998e-13
30has Value4.04799e-24
40has Value1.34758e-36
50has Value3.7019e-50
60has Value1.38556e-64
70has Value9.85586e-80
80has Value1.68916e-95
90has Value8.33228e-112
100has Value1.3583e-128
Node has 0value 1.00028
Total Value is 10.0028
[cs5440s23@oscar ~]$

```