MPI Collective Comm. & Comp. Lib. Calls

Example 2.1

Write MPI program to broadcast message "Hello world" to all the process.

Example 2.2

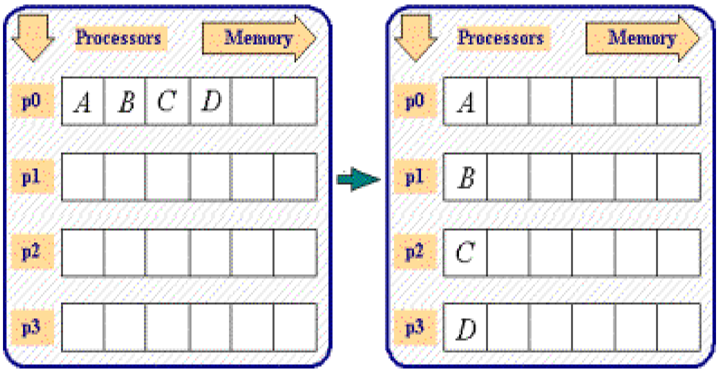
Write MPI program to scatter n integers on p processes (Assume that n is divisible by p) .

Objective

Write a MPI program to scatter an integer array **A** of dimension *n* on *p* processors of IBM AIX cluster using **MPI_Scatter** communication library calls.

Description

Assume that *n* is multiple of *p* and index of the array **A** starts from 0 (as in C-Program). Processor with rank *k, k* =0,1, ..., *p*-1 gets *n/p* values starting from **A**[*i*], *i* = *k***n/p* to (*k*+1)**n/p* - 1*.*



Input

Process with rank *0* reads the input data.You have to adhere strictly the following format for the input file.

#Line 1 : *Size of the array*

#Line 2 : *data* in order, this means that the data of second entry of the input array **A** follows the first and so on.

24
2 10 3 4 23 14 4 6 8 32 63 86 12 8 3 9 13 4 14 16 18 2 9 86

Output

Each process prints its own *n*/*p* elements of array **A.**

Example 2.3

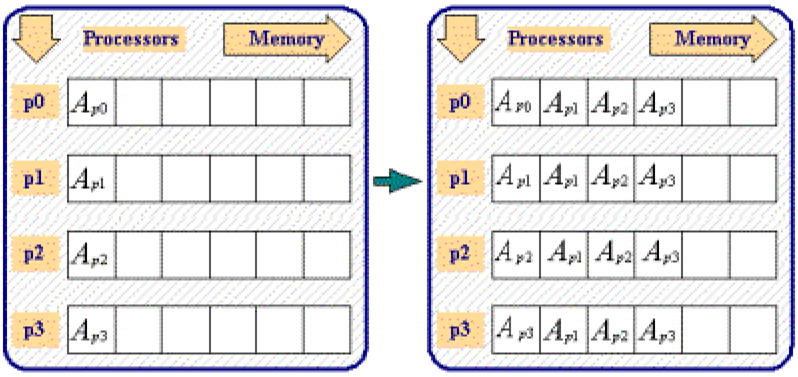
Write MPI Program to gather n integers from each process and obtain the resultant data on each process.

Objective

Write a MPI program to gather values from each process and store the output values in the global integer array **A** of dimension *n* on *p* process of cluster. *Use* **MPI_Allgather** Communication library call in the *program*.

Description

Assume *n* be the dimension of an integer input array **A**, output array **B**, and *p be the* processes used in the parallel program. The integer array **A** [i] of size *n/p* with known *n/p* values where varies from 0 to *n/p* - 1 is given as additional input on processor with rank *k* (*k* = 0,1, ... *p*-1). Each process reads respective input array of size *n/p* from the given input file. You have to adhere strictly the following format for the input file.



Processors
Memory
p0
p1
p2
p3
A_{p0}
A_{p1}
A_{p2}
A_{p3}
Processors
Memory
p0
p1
p2
p3
A_{p0} A_{p1} A_{p2} A_{p3}
A_{p1} A_{p1} A_{p2} A_{p3}
A_{p2} A_{p1} A_{p2} A_{p3}
A_{p3} A_{p1} A_{p2} A_{p3}

Input

Each process *k* reads the input file.You have to adhere strictly the following format for the input file.

#Line 1 : *Size of the local array*

#Line 2 : *data* in order, this means that the data of second entry of the input array **A** follows the first and so on.

A sample input file is given below.

Input file for process 0
6
21 18 54 24 33 69

- **Output**

  each process prints the output array **A.**

Example 2.4

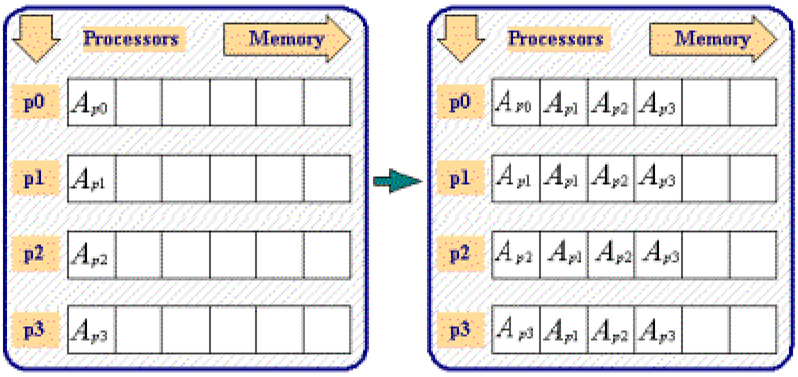
Write MPI Program to perform all-to-all communication of n integers on p process. (Assume that the each process has n integers where n is divisible by p).

Objective

Write a MPI program to gather values from each process and store the output values in the global integer array **A** of dimension *n* on *p* process of cluster. Use *MPI_Allgather* Communication library call in the program.

Description

Assume *n* be the dimension of an integer input array **A**, output array **B**, and *p* be the processes used in the parallel program. Each process reads input integer array of size *n* from the given input file. Assume that *n* is divisible by *p*


Processors
Memory
p0
A_{p0}
p1
A_{p1}
p2
A_{p2}
p3
A_{p3}
Processors
Memory
p0
A_{p0} A_{p1} A_{p2} A_{p3}
p1
A_{p1} A_{p1} A_{p2} A_{p3}
p2
A_{p2} A_{p1} A_{p2} A_{p3}
p3
A_{p3} A_{p1} A_{p2} A_{p3}

Input

Each process *k* reads the input file. You have to adhere strictly the following format for the input file.

#Line 1 : *Size of the local array*

#Line 2 : *data* in order, this means that the data of second entry of the input array **A** follows the first and so on.

A sample input file is given below. For convenience sake, 4 processes are used in this example.

Input file for process 0
16
21 6 8 45 54 24 33 72 84 12 25 31 46 16 94 65

Output

Each process prints the output array **B.**

Example 2.5

Write MPI Program to compute the sum of q on p process and process with rank 0 prints the resultant sum. (Assume that the each process has q integers where q is equal to n/p and n is total number of integers on each process).

Objective

Write a MPI program to find sum of *q* integers, on *p* processors of message passing cluster, using **MPI** collective communication and computation library call.

Description

MPI collective communication and computation library call, **MPI_Reduce** is used.

Input

For input data, let each process use its identifying number, i.e. the value of its rank. For example, process with rank *0* uses the input integer r 0, process with rank *1* uses the input integer number 1, etc.

Output

Process with rank 0 prints the final sum.

Example 2.6

Write MPI program to compute the sum of q integers on p process and each process prints the resultant sum. (Assume that the each process has q integers and q is equal to n/p and n is total number of integers)

Objective

Write a MPI program to find sum of *q* integers, on *p* processors of message passing cluster, using **MPI** collective communication and computation library call.

Description

In this example **MPI** collective communication and computation library call, **MPI_Allreduce** is used.

Input

For input data, let each process use its identifying number, i.e. the value of its rank. For example, process with rank *0* uses the input integer 0, process with rank *1* uses the input integer number 1, etc.

Output

Each process prints the final sum.