Code:

1. ScatterGather.java

ScatterGather.java 9+ ●

ScatterGather.java > ScatterGather > main(String[])

```java
import mpi.MPI;

import java.util.Scanner;

import mpi.*;

public class ScatterGather {
    Run | Debug
    public static void main(String[] args) throws Exception{
        MPI.Init(args);
        //  Get rank of each process and size of communicator
        int rank = MPI.COMM_WORLD.Rank();
        int size = MPI.COMM_WORLD.Size();

        int unitsize = 5;
        int root = 0;
        int send_buffer[] = null;
        //  1 process is expected to handle 4 elements
        send_buffer = new int [unitsize * size];
        int recieve_buffer[] = new int [unitsize];
        int new_recieve_buffer[] = new int [size];

        //  Set data for distribution
        if(rank == root) {
            int total_elements = unitsize * size;
            System.out.println("Enter " + total_elements + " elements");
            for(int i = 0; i < total_elements; i++) {
                System.out.println("Element " + i + "\t = " + i);
                send_buffer[i] = i;
            }
        }

```

ScatterGather.java 9+

ScatterGather.java > ScatterGather > main(String[])

```
32          //  Scatter data to processes
33          MPI.COMM_WORLD.Scatter(
34              send_buffer,
35              0,
36              unitsize,
37              MPI.INT,
38              recieve_buffer,
39              0,
40              unitsize,
41              MPI.INT,
42              root
43          );
44
45          //  Calculate sum at non root processes
46          //  Store result in first index of array
47          for(int i = 1; i < unitsize; i++) {
48              recieve_buffer[0] += recieve_buffer[i];
49          }
50          System.out.println(
51              "Intermediate sum at process " + rank + " is " + recieve_buffer[0]
52          );
53
54
55          //  Gather data from processes
56          MPI.COMM_WORLD.Gather(
57              recieve_buffer,
58              0,
59              1,
60              MPI.INT,
61              new_recieve_buffer,
62              0,
63              1,
64              MPI.INT,
65              root
66          );
67
68          //  Aggregate output from all non root processes
69          if(rank == root) {
70              int total_sum = 0;
71              for(int i = 0; i < size; i++) {
72                  total_sum += new_recieve_buffer[i];
73              }
74              System.out.println("Final sum : " + total_sum);
75          }
76
77          MPI.Finalize();
78      }
79  }
```

Output:

1. Compilation and Execution