Lab 2: Collective Communication

1. Compile and run the “Pi calculation” program as follows. Try with different number of intervals and number of processes

```
#include "mpi.h"
#include <stdio.h>
#include <math.h>

int main(int argc, char *argv[])
{
	int n, myid, numprocs, i;
	double PI25DT = 3.14159265358979323846264643;
	double pi, w, sum, x, a;
	MPI_Init(&argc, &argv);
	MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
	MPI_Comm_rank(MPI_COMM_WORLD, &myid);
	if (myid == 0)
	{
		printf("Enter the number of intervals: (0 quits) ");
		fflush(stdout);  /* Force screen output now */
		scanf("%d", &n); /* Only process 0 can use scanf */
	}
	MPI_Bcast(&n, 1, MPI_INT, 0, MPI_COMM_WORLD);
	if (n > 0)
	{
		w = 1.0 / (double)n;
		sum = 0.0;
		for (i = myid + 1; i <= n; i += numprocs)
		{
			x = w * ((double)i - 0.5);
			sum += w*4.0 / (1.0 + x * x);
		}
		MPI_Reduce(&sum, &pi, 1, MPI_DOUBLE, MPI_SUM, 0,
			MPI_COMM_WORLD);

		if (myid == 0)
			printf("pi is approximately %.16f, Error is %.16f\n",
				pi, fabs(pi - PI25DT));
	}
	MPI_Finalize();
}
```

2. Modify the “integersum.c” from the previous lecture as follows:

- Master process asks for 2 user inputs: left and right
- Master process broadcasts left and right to all slaves
- Replace MPI_Send/MPI_Recv with MPI_Reduce operation to combine partial results into the final result