```cpp
//------------------------------------------------------------------------
// topology 1-dimensional
//------------------------------------------------------------------------
#if (defined __linux__) || (defined _AIX) || (defined __APPLE__)
    #include <sys/types.h>
    #include <sys/stat.h>
    #include <unistd.h>
#elif (defined _WIN32) || (defined _WIN64)
    #include <conio.h>
    #include <direct.h>
#endif

#include <mpi.h>
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
using namespace std;

int main(int argc, char** argv)
{
    int i, n = 0, rank, size, dest, src, mess1 = 0, mess2 = 0, mysumm = 0;
    int dims[1], periods[1];
    MPI_Request request;
    MPI_Status status;
    MPI_Comm mycomm_cart; // communicator

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    if (rank == 0) {
        printf("\nThere are %d processes.\n", size);
        cout << "Select how many times messages, i.e., ranks, should be shifted"
            << " in the direction of topology!\nIt should be n >= 0 and"
            << " n < " << size << ".\n  n = ";
        cin >> n;
        fflush(stdout);
    }
    MPI_Bcast(&n, 1, MPI_INT, 0, MPI_COMM_WORLD);
    dims[0] = size;
    periods[0] = 1;

    MPI_Cart_create(MPI_COMM_WORLD, 1, dims, periods, true, &mycomm_cart);

    mess1 = rank;

    for (i = 0; i < n; i++) {
        MPI_Cart_shift(mycomm_cart, 0, 1, &src, &dest);
        printf("\n%d -> %d", src, rank);
        MPI_Issend(&mess1, 1, MPI_INT, dest, 0, mycomm_cart, &request);
        MPI_Recv(&mess2, 1, MPI_INT, src, 0, mycomm_cart, &status);
        printf("\nprocess %d: message = %d.", rank, mess2);
        mysumm += mess2;
        MPI_Wait(&request, &status);
        mess1 = mess2;
    }
    printf("\nprocess %d: summ of messages = %d\n", rank, mysumm);
    MPI_Finalize();
    return 0;
}
```