1. Write a MPI program to multiply the respective elements of two arrays.

Assume that the number of elements in each array is N and is evenly divisible by the number of processes. Also assume that N is greater than the number of processes that you create.
The root process reads these two array elements. The other processes will get the required elements of arrays from the root that they need to handle.
All the processes will involve in multiplying the respective elements that they need to handle including root.
All the processes will give their product elements to the root process. Let each process print the product elements computed by them.
Root process prints the product array after collecting the product elements.
Do not use point to point communication APIs in your program. Also show the amount of time taken by the program to perform this operation.

2.Write an efficient CUDA kernel to meet the following:
It is required to generate the row sums and the column sums of a given matrix *A* of size *N* X *N*.

For Eg, if the given matrix A is of size 3 X 3:

Matrix **A** =	1	2	3
	4	5	6
	7	8	9

Expected output is

Row sums in Array **B** =	6	15	24
Column sums in Array **C** =	12	15	18

A verification is required for how your kernel will be functioning for the given matrix A. Use any two threads to show how the parallelism is observed in your kernel for the given data.

tid = 0	tid = 1