1. **Design and implement a C program to find the sum of the element in an array using MPI.**

Hint: The main master process reads the numbers and sends random numbers from the array to sub processes for parallel execution. Each sub process calculate their partial sums and then finally, the master process collects and calculates the sum of these partial sums to return the total sum of the array.

**You are free to choose the process count, communication scenario, and array elements.**

**Draw the design in a paper and answer the following questions:**

**Design: 8 marks**

1. Draw the communication world diagram showing process numbers and the communication link between them. – **3 marks**
2. What type of communication do you select? (point-to-point or collective) - **2 marks**
3. What type of communication mode is planned? (Blocking or non-blocking) **– 2 marks**
4. Give justification for your selection of the above **- 1 mark**

**Implementation and Output: 12 marks**

* Fully correct in logic and full output: 12 marks
* Partially implemented or partial output: 6 marks
* Partially implemented or no output : 4 marks
* Implementation or code error : 0 marks

**Total: 20 marks**

**Sample Output format**:

NB: Print the partial sum calculate by each process and final sum of the array calculated by root process

**Example:**

Enter the array elements: 20

Sum calculated by root process: 10

Sum calculated by Process 1: 40

Partial sum returned from Process 1: 40

Sum calculated by Process 2: 50

Partial sum returned from Process 2: 50

Sum of array: 100

1. Design and implement a C program that takes data from process zero and sends it to all of the other processes by sending it in a ring. Suppose there are ‘n’ processes, process i should receive the data from user and send it to process i+1, until the last process n receives the data. Finally, process i should receive back the data from process n.

**Draw the design in a paper and answer the following questions:**

**Design: 8 marks**

1. Draw the communication world diagram showing process numbers and topology. – **3 marks**
2. What type of communication do you select? (point-to-point or collective) - **2 marks**
3. What type of communication mode is planned? (Blocking or non-blocking) **– 2 marks**
4. Give justification for your selection of the above **- 1 mark**

**Implementation and Output: 12 marks**

* Fully correct in logic and full output: 12 marks
* Partially implemented or partial output: 6 marks
* Partially implemented or no output : 4 marks
* Implementation or code error : 0 marks

**Total: 20 marks**

**Sample Output format**:

Let the number of process be 3 and assume that the data being send is a token with name “CSE312”. At the beginning, Process zero reads the data from the user. Print the rank and data received by each process.

**Example:**

Enter the data:

CSE312

Process 0 receives CSE312

Process 1 receives CSE312

Process 2 receives CSE312

Process 0 receives CSE312