Q1.a (2 marks). What is the maximum possible speedup possible if the hardware has 20 stages ?

Q. 1 b. (1 marks) True/False. Multithreading solves bw problem but exacerbates latency problem

Q.1 c. (1 marks) True/False. EREW model is the strongest PRAM architecture.

Q. 2 (2 marks). Following MPI program with 2 processes has deadlock. State the reason of the deadlock.

The arguments are in this order : memory address, tag, process_id

```
     P0                   P1
 receive(&a, 1, 1);   receive(&a, 1, 0);
 send(&b, 1, 1);      send(&b, 1, 0);
```

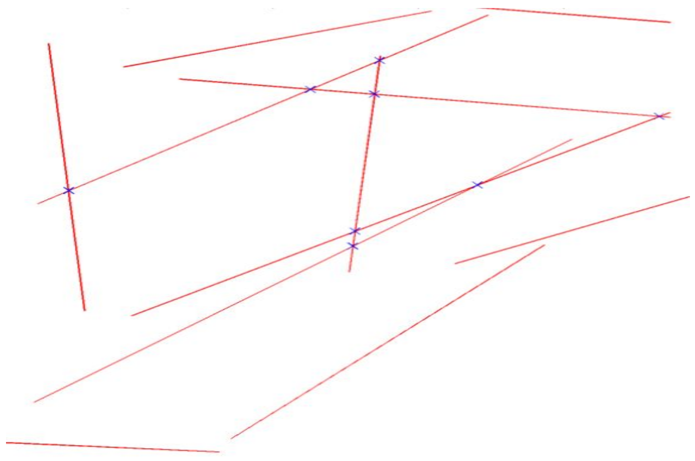
Q3 (2 marks). For the following block of code running on two processors P1 and P2

```
if(A == 4)
printf("hi");
else
printf("bye");
```

Describe how will the code block execution by P1 and P2 vary in SIMD and MIMD architecture ?

Q4. (3 marks). Prove that the diameter of a hypercube is O(log p) where p is the number of processors.

Q5. (4 marks). Given a collection of line segments as input and a function bool isIntersecting(line_segment seg1, line_segment seg2) which returns true/false based on whether seg1 and seg2 intersects. Here line_segment contains start vertex and end_vertex. Provide a parallel algorithm and state its time complexity for counting the number of segment intersections by p processes.



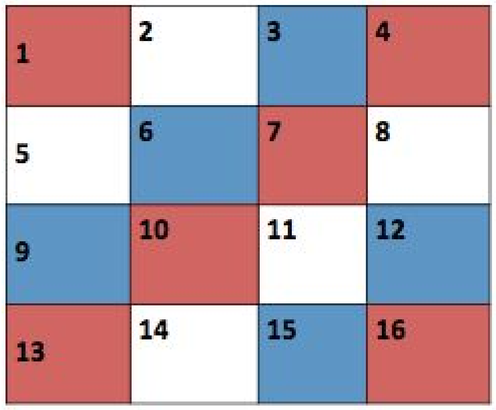
Q6) (2 marks) Given below is a function *insertElement* to insert a value to a shared array *arr*. This function does not work correctly in a multi-threaded environment. State potential bugs and a method to make the code thread-safe.

```
#define MAX 1000

int arr[MAX];
int position = 0;

int main()
{
  insertElement(10);
  return 0;
}
```

```
void insertElement(int info)
{
  if(position < MAX) {
    arr[position] = info;
    position++;
  } else {
    printf("arr is full");
  }
}
```

Q7.(6 marks) Write a pseudo-code (algorithm) using MPI to add the elements of an **m** by **n** matrix with the help of **p** processes. Each process gets matrix elements in a cyclic fashion as described below.

Here is an example: A input matrix with each cell having a number to be divided among 3 processes as shown:



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Processor P0 should add red cells, P1 should add white cells and P2 should add blue cells.

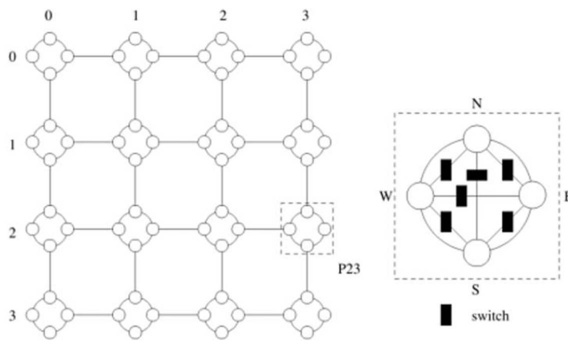
P0 adds the red cells locally = 1 + 4 + 7 + 10 + 13 + 16 = 51
P1 adds the white cells locally = 2 + 5 + .. + 14 = 40
P2 adds the blue cells locally = 3 + 6 + .. + 15 = 45

The final output should be the summation of the local sums 51, 40 and 45 to yield 136.
Assume:
1<p<8
row<10
col<10

Q8. Book question (chapter 2, Q 2.16)

2.16 [MKRS88] A $\sqrt{p} \times \sqrt{p}$ ***reconfigurable mesh*** consists of a $\sqrt{p} \times \sqrt{p}$ array of processing nodes connected to a grid-shaped reconfigurable broadcast bus. A 4 × 4 reconfigurable mesh is shown in **Figure 2.35**. Each node has locally-controllable bus switches. The internal connections among the four ports, north (N), east (E), west (W), and south (S), of a node can be configured during the execution of an algorithm. Note that there are 15 connection patterns. For example, {SW, EN} represents the configuration in which port S is connected to port W and port N is connected to port E. Each bit of the bus carries one of ***1-signal*** or ***0-signal*** at any time. The switches allow the broadcast bus to be divided into subbuses, providing smaller reconfigurable meshes. For a given set of switch settings, a ***subbus*** is a maximally-connected subset of the nodes. Other than the buses and the switches, the reconfigurable mesh is similar to the standard two-dimensional mesh. Assume that only one node is allowed to broadcast on a ***subbus*** shared by multiple nodes at any time.

Figure 2.35. Switch connection patterns in a reconfigurable mesh.



Determine the bisection width, the diameter, and the number of switching nodes and communication links for a reconfigurable mesh of $\sqrt{p} \times \sqrt{p}$ processing nodes. What are the advantages and disadvantages of a reconfigurable mesh as compared to a wraparound mesh?

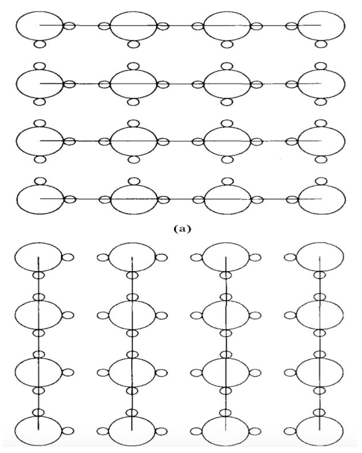
If each processor has 1 element, provide a parallel algorithm to find the minimum element.

<u>More info:</u>

When all switches are closed, all n^2 processors are connected by the bus.

If all processors disconnect the switch on their north, then we obtain row buses.

Column buses are obtained by having each processor disconnect the switch on its east.



Q9. LU decomposition : How to parallelize this problem ?
Identify the tasks that could be executed in parallel and the task dependencies. Also, show how to efficiently map these tasks to processes.

LU decomposition (where 'LU' stands for 'lower upper') decomposes a matrix as the product of a lower triangular matrix and an upper triangular matrix.
A = LU

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}$$

Attachment in d2l.mu.edu