

**GRIFFITH COLLEGE DUBLIN**

**QUALITY AND QUALIFICATIONS IRELAND  
EXAMINATION**

**POSTGRADUATE DIPLOMA IN SCIENCE IN BIG DATA MANAGEMENT  
AND ANALYTICS**

**PARALLEL AND DISTRIBUTED PROGRAMMING  
Module Code: PGDBD-PDP**

**MASTER OF SCIENCE IN BIG DATA MANAGEMENT AND ANALYTICS**

**PARALLEL AND DISTRIBUTED PROGRAMMING  
Module code: MSCBD-PDP**

**ONLINE EXAMINATION**

**Lecturer(s):**

**Osama Abushama**

**External Examiner(s):**

**Dr Joseph Rafferty**

**Date: 25<sup>th</sup> January 2022**

**Time: 2.15-5.15**

**THIS PAPER CONSISTS OF FIVE QUESTIONS  
FOUR QUESTIONS TO BE ATTEMPTED  
ALL QUESTIONS CARRY EQUAL MARKS**

## **HONOUR CODE**

By submitting my exam script, I certify that my answers contained in this Examination Script document are entirely composed of my own original work.

During the exam period, which began when I received the exam paper document, I did not work with anyone else on the exam or discuss the examination with anyone else.

I did not access any unauthorised material or copy and hold out as my own any material belonging to or produced by another person.

I understand that failure to adhere to these instructions shall be an Honour Code violation.

Violation of the Honour Code will result in being charged with academic misconduct.

### QUESTION 1

- (a) Explain four different types of MPI send? Describe each type characteristics.  
(10 marks)
- (b) Explain what condition prevent using OpenMP parallel loops?  
(5 marks)
- (c) Write an OpenMP C program matrix multiplication  $A = B * C$ , where B and C are diagonal matrices?  
(5 marks)
- (d) Modify Part 3 if the required multiplication is  $B = B * C$ , what conditions need to take into consideration if any for this multiplication to be correct.  
(5 marks)
- Total (25 marks)**

### QUESTION 2

- (a) Write an OpenMP C program to compute the average of nxn matrix A rows and save the average of each row in a one dimensional array. Your code need to be optimal.  
(10 marks)
- (b) Re-write the same task in part 1 using MPI.  
(15 marks)
- Total (25 marks)**

### QUESTION 3

- (a) Explain main advantages of using ReentrantLocks over Semaphores?  
(5 marks)
- (b) Write a Java class to control access to a classroom of 100 students using semaphores, with guarantee that first come student allowed to access before late coming students, your class should have two methods EnterClass and LeaveClass.  
(15 marks)
- (c) Write a main program to test part 2, generating 30 students every 5 minutes. You do not need to write the Student thread class.  
(5 marks)
- Total (25 marks)**

#### QUESTION 4

- (a) Class **MyList** given below uses a binary **Semaphore** to make the methods **set** and **inc** thread safe. The solution uses *coarse-grained* synchronization. Re-write the class using semaphores to provide a *fine-grained* synchronized solution that makes the class thread safe.

```
class MyList{
    private int[] list = new int[100];
    private Semaphore lock = new Semaphore(1);
    MyList(){
        for(int j = 0; j < 100; j++) list[j] = (int)(Math.random()*100);
    }
    public void set(int x, int ind){ //assume 0 <= ind < 100
        try{lock.acquire();}
        catch(InterruptedException e){}
        list[ind] = x;
        lock.release();
    }
    public void inc(int x, int ind){//assume 0 <= ind < 100
        try{lock.acquire();}
        catch(InterruptedException e){}
        list[ind] += x;
        lock.release();
    }
}
```

(15 marks)

- (b) Explain why class Q4 below is not thread safe. Explain why it is not thread safe, Re-write it to make it thread safe.

```
class Q4{
    private boolean f[] = new boolean[20];
    public Q4(){ for(int j = 0; j < 20; j++) f[j] = true;}
    public void negate(int a, int b){
        for(int j = a; j < b; j++)f[j] = !f[j];
    }
    public void swap(int a, int b){
        boolean temp = f[a];
        f[a] = f[b];
        f[b] = temp;
    }
}
```

(10 marks)

**Total (25 marks)**

### QUESTION 5

- (a) Write a class called `Buffer` that holds a single integer value. The class has two public methods called `read()` that returns the current value of the buffer and `write(k)` that writes integer `k` to the buffer. A thread invoking `read` must wait for a value to be written to the buffer. Once the item in the buffer is read it is cleared to allow a writer thread write. If the buffer contains a value a writer thread must wait for the current value of the buffer to be read by another thread.
- (b) Explain why the use of static variables and instance variables in the definition of a class complicates things when writing methods that support concurrent access for threads.
- (c) With reference to the `Lock` class explain the semantics of the `tryLock()` method.

(15 marks)

(5 marks)

(5 marks)

**Total (25 marks)**