Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

2020 SAMPLE PAPER BASED ON SEMESTER 2 EXAMINATIONS 2018/2019

MODULE: CA4006 - Concurrent and Distributed Programming

PROGRAMME(S):

CASE	BSc in Computer Applications (Sft.Eng.)
CPSSD	BSc in ComputationalProblem Solv&SW Dev.
ECSAO	Study Abroad (Engineering & Computing)

YEAR OF STUDY: 4,O

EXAMINER(S):

Dr. Rob Brennan	(Internal)	(Ext:6008)

TIME ALLOWED: ~~3 Hours~~

INSTRUCTIONS: **Answer 4 questions. All questions carry equal marks.**

Note on modifications to paper for open book format: New text is **bold**. Deleted text is ~~strikethrough~~.

~~**PLEASE DO NOT TURN OVER THIS PAGE UNTIL YOU ARE INSTRUCTED TO DO SO.**~~
~~The use of programmable or text storing calculators is expressly forbidden.~~
~~Please note that where a candidate answers more than the required number of questions, the examiner will mark all questions attempted and then select the highest scoring ones.~~
New paper instructions to be provided.

There are no additional requirements for this paper.

QUESTION 1 ***[TOTAL MARKS: 25]***

Q 1(a) **[12 Marks]**

Distinguish clearly between processes and threads. Differentiate, using ~~diagrams~~ **examples** and definitions the differences between Multithreading and Multitasking. **Explain in what circumstances you would use each one and why.**

Q 1(b) **[13 Marks]**

[3 marks]

(i) Define clearly, in words and as a formula, what is meant by Amdahl's law. **Give an example of when you would use it.**

[5 marks]

(ii) For a piece of code you have written, you know that memory operations currently take 30% of execution time. A new widget speeds up 80% of memory operations by a factor of 4 and a second new widget speeds up 1/2 the remaining 20% by a factor or 2. Using Amdahl's law calculate the total speed up from these two widgets.

[5 marks]

(iii) Latency may be loosely defined as the time interval between the stimulation and response, or, from a more general point of view, as the time delay between the cause and the effect of some physical change in the system being observed. ~~Interpret latency in terms of Amdahl's law, expressing it in terms of the speedup and the parallel fraction. Use this to show that, in terms of the speedup, things can only get so fast, but they can get arbitrarily slow.~~ **Explain with an example the relationship between latency and Amdahl's law. What are the consequences for parallel system design and deployment?**

[End of Question1]

QUESTION 2 ***[TOTAL MARKS: 25]***

Q 2(a) **[8 Marks]**

Explain clearly the differences between semaphores and monitors. ~~Write a short code fragment showing how to guarantee Mutual Exclusion for *N* processes using semaphores in C. You may assume the existence of semaphores (with notation `sem`) and operations on them in C. Using the semaphore invariant outline a simple proof that Mutual Exclusion is satisfied.~~ **What would be the differences between implementing mutual exclusion for N processes using semaphores or monitors? Which is the most effective method in your opinion and why?**

Q 2(b) **[9 Marks]**

~~Describe fully the algorithm for the Reader-preference solution of the Readers-Writers problem using semaphores. Write fully commented C code that implements the algorithm. You may assume the existence of semaphores (with notation `sem`) and operations on them in C as well as `Read_Database()` and `Write_Database()`.~~

Examine the code below:

```
int M:= 20; int N:= 5;
int nr:=0;
sem mutexR := 1; sem rw := 1
process one (i:= 1 to M) {
   while (1)      {
      wait (mutexR);
      nr := nr + 1;
      if (nr = 1)
            wait (rw);
      signal (mutexR);
      Read_Database ( );
      wait (mutexR);
      nr := nr – 1;
      if (nr = 0 )
            signal (rw)
            signal (mutexR);
   }
}
process two(i:=1 to N) {
   while (1)
      wait (rw);
       Update_Database ( );
      signal (rw);
   }
```

```
}
```

What standard concurrency algorithm is this? Describe the problem it is solving and both the advantages and disadvantages of this approach. Document your assumptions and write pseudo-code for a revised system to deal with a deployment where the database is never updated during runtime but you may have millions of read requests.

Q 2(c) **[8 Marks]**

Describe fully **the advantages and disadvantages of** the algorithm for Ballhausen's solution to the problem of Reader-Preference ~~in the Readers-Writers algorithm using semaphores in Q2(b) above. Write carefully commented C code that implements the algorithm.~~ **Describe a scenario in which you would recommend deploying Ballhausen's solution and explain why it is appropriate.**

[End of Question2]

QUESTION 3 *[TOTAL MARKS: 25]*

Q 3(a) [6 Marks]

Explain the difference between Low Level Concurrency Objects and High Level Concurrency Objects in Java **using an example from your first continuous assessment project**. Why would you use the latter rather than the former?

Q 3(b) [8 Marks]

~~Write fully explained and commented code for the sleeping barber problem using monitors in C where the barber and customers are interacting processes, and the barber shop is the monitor in which they interact. You are only required to implement the body of the monitor, not the main code that uses it.~~

Write fully explained and commented code for the following problem using monitors in C:

Many application processes communicate by taking turns in a single shared fixed length queue. There is a single network I/O process that can send items from the application process at the head of the queue. After finishing sending an item, the I/O process dismisses the head application process and checks the queue to see if there are any waiting application processes. If there is one then the I/O process moves it to the head of the queue and starts to send its data, if there are none, the I/O process yields to the operating system.

Each time an application process wants to communicate, it checks to see what the I/O process is doing. If it is sleeping then it wakes it up and moves to the head of the queue. If the I/O process is working then the application process joins the queue if it is not full, otherwise it sleeps.

Q 3(c) [11 Marks]

Write code for the above ~~sleeping barber~~ problem using `lock` and `condition` objects in Java to implement a **`NetworkIOSystem`**~~`Barbershop`~~ class. The **IO process**~~barber~~ and **applications**~~customers~~ are interacting processes, and the network I/O system~~barber shop~~ is the monitor in which they interact. You are only required to implement the body of the monitor, not the main code that uses it.

[End of Question3]

QUESTION 4 ***[TOTAL MARKS: 25]***

Q 4(a) **[6 Marks]**

Describe the SPMD parallel program structure pattern and give examples of (i) a type of problem that it is suitable for solving and (ii) a challenge or limitation of the pattern and (iii) a programming framework designed to support SPMD**, explaining how and why it is supported**.

Q 4(b) **[14 Marks]**

Given this pseudo-code for Mandelbrot set calculation, write carefully commented code to parallelise the calculation using the C language and OpenMP using 10 processors. Calling the `plot()` function should create a 1920 by 1080 pixel 2-D array that will eventually be written to the screen by other code that you do not have to provide.

```
maxit = 1000
for each pixel (px, py) {
     x0 = scaled x coordinate of pixel (scaled to lie in
the Mandelbrot X scale (-2.5, 1));
     y0 = scaled y coordinate of pixel (scaled to lie in
the Mandelbrot Y scale (-1, 1));
     x = 0.0;
     y = 0.0;
     it = 0;
     while ( it < maxit AND x*x + y*y ≤ 2*2 ) {
          xnew = x*x - y*y + x0;
          ynew = 2*x*y + y0;
          x = xnew;
          y = ynew;
          iteration = iteration + 1;
     }
     plot(px, py, it);
}
```

Figure Q4. Mandelbrot Set Calculation

Q 4(c) **[5 Marks]**

Please identify your task or data partitioning strategy for your program from 4(b) and explain your rationale for using it i.e. what strategy have you adopted and what factors did you consider when selecting it?

[End of Question4]

QUESTION 5 ***[TOTAL MARKS: 25]***

Q 5(a) **[14 Marks]**

Compare and contrast two types of distributed systems middleware from the perspectives of (i) network transparency, (ii) communications persistence and delivery semantics (iii) key features. ~~Include an annotated diagram showing the overall middleware architecture in each case.~~ Give an example deployment where each type of middleware would be an appropriate choice **and explain why**.
If you wanted to build an enterprise application for monitoring transactions in financial systems, what type of middleware would you recommend using and why?

Q 5(b) **[11 Marks]**

Create a set of 3 UML interaction diagrams for a reliable asynchronous RPC service that assures "at least once" delivery semantics for its clients. Make sure the service can handle at least 3 failure modes and create one diagram per mode. Each diagram should show both failure and how the system recovers. Use a separate lifeline for the server hander and dispatcher processes. Add all required message parameters that are needed to assure the “at least once” semantics. Document any assumptions you make.

[End of Question5]

[END OF EXAM]