Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

AUGUST/RESIT 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. Martin Crane	(Internal)	(Ext:8974)
Dr. Rob Brennan	(Internal)	(Ext:6008)
Dr. Hitesh Tewari	(External)	External
Prof. Brendan Tangney	(External)	External

TIME ALLOWED: 3 Hours

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

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.

There are no additional requirements for this paper.

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

Q 1(a) **[4 Marks]**

Define carefully what is meant by *safety* and *liveness* in concurrent programming.

Q 1(b) **[8 Marks]**

Write carefully commented code in C implementing Dekker's algorithm and explain how it overcomes problems of contention.

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

Implement in Java of Dekker's algorithm, carefully commenting your code. Indicate with appropriate comments the critical and non-critical sections.

[End of Question1]

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

Q 2(a) **[4 Marks]**

Explain the differences between semaphores and monitors in C.

Q 2(b) **[7 Marks]**

Give a high level description of the algorithm for the solution of the Producer-Consumer problem for infinite buffers using semaphores. Write C code that implements the algorithm.

Q 2(c) **[14 Marks]**

Using monitors, write code in Java to implement a solution to the Producer Consumer Problem. Your implementation should provide the following classes:

1. `class Buffer` which has methods `get`( ) and `put`( ) to retrieve and append integers onto a buffer. (The `Buffer` need only store one integer at a time).
2. `Producer` and `Consumer` classes which utilise the `put( )` and `get( )` methods (above) respectively.
3. A `ProducerConsumerTest` class to test the classes above by instantiating producer and consumer objects.

[End of Question2]

QUESTION 3 *[TOTAL MARKS: 25]*

Q 3(a) [5 Marks]

Define what is meant by a thread-safe Java class giving and give an example of a type of Java object that is always thread-safe. Briefly describe two steps you could take to ensure a Java class is thread-safe.

Q 3(b) [9 Marks]

Using a diagram illustrate how, why and where the following Java code operates poorly in a concurrent environment.

```
public class SynchronizedHash implements Servlet {
@GuardedBy("this") private Object lastObject;
@GuardedBy("this") private int lastCode;
public synchronized void service(ServletRequest req,
     ServletResponse resp) {
          Object i = extractFromRequest(req);
          if (i.equals(lastObject))
               encodeIntoResponse(resp, lastCode);
          else {
               int code = i.hashCode());
               lastObject = i;
               lastcode = code;
               encodeIntoResponse(resp, code);
          }
     }
}
```

Figure Q3 (b) Poor Concurrent Java Code

Q 3(c) [11 Marks]

Carefully re-write and comment the code given in Q2(b) so that it is more performant.

[End of Question3]

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

Q 4(a) [8 Marks]

In the context of Java’s Remote Method Invocation (RMI), define what is meant by stubs and skeletons, illustrating your answer with a diagram showing the RMI architecture. Show how the skeleton and stub may be generated.

Q 4(b) [17 Marks]

A Remote Method Invocation (RMI) Interface to allow a client to invoke a command to find the factorial of a number on a remote machine is shown in Figure Q 4. You are required to implement the remote interface, develop the server and develop a client that invokes the remote method fact. Your code should be fully commented and should document the function of each major component.

```
import java.rmi.*;
public interface RemoteInterface extends Remote
{
     public int fact(int x) throws Exception;
}
```

Figure Q 4. Factorial.java

[End of Question4]

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

Q 5(a) **[13 Marks]**

“The web services architecture draws heavily on RPC and DCE concepts from the early 1990s.” Discuss this statement with respect to the following topics: (i) stubs and skeletons (ii) Interface Definition Languages (IDLs) (iii) common services for distributed systems (iv) communication protocols and (v) naming. Include an architectural diagram of each system.

Q 5(b) **[12 Marks]**

Part of a Java Interface SensorReading.java to return the location and the associated Carbon Dioxide level of a room is shown in Figure 5. Using Java Web Services, implement the following components of this interface: the Service Endpoint Interface (SEI), the Service Implementation Bean (SIB) and the Endpoint Publisher. You should fully comment your code.

```
public String getLocation (String Room){
  // implementation omitted
}
public String getCO2Level (String Room) {
/  / implementation omitted
}
```

Figure Q5

[End of Question5]

[END OF EXAM]