DUBLIN CITY UNIVERSITY

SEMESTER 2 EXAMINATIONS 2015/2016

MODULE: CA4006 – Concurrent and Distributed Programming

PROGRAMME(S):

CASE	BSc in Computer Applications (Sft.Eng.)
ECSAO	Study Abroad (Engineering & Computing)
SHSAO	Study Abroad (Science & Health)

YEAR OF STUDY: 4,O

EXAMINERS:

Dr. Martin Crane (Ext:8974)
Dr. Ian Pitt

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.

Requirements for this paper (Please mark (X) as appropriate)

	Log Tables		*Thermodynamic Tables*
	Graph Paper		*Actuarial Tables*
	Dictionaries		*MCQ Only – Do not publish*
	Statistical Tables		*Attached Answer Sheet*
	Bible		

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

Q 1(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.

Q 1(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()`.

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

Describe fully the algorithm for Ballhausen's solution to the problem of Reader-Preference in the Readers-Writers algorithm using semaphores in Q1(b) above. Write carefully commented C code that implements the algorithm.

[End of Question 1]

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

2(a) **[11 marks]**

In the context of concurrency, what is meant by a *programming model*? How does it relate to the machine model? Explain, using diagrams, the three principal programming models in a concurrent system.

2(b) **[10 marks]**

In the context of Structured Pier to Pier (P2P) architectures, the Chord algorithm is used for managing nodes logically organized in a ring and the routing of enquiries about data items that are mapped to those entities nodes. Briefly describe the steps in the Chord algorithm. One such ring is given in Figure Q.2 below with the finger tables for nodes. Complete the finger tables for nodes 10, 12 and 15 in Figure Q.2. Give the steps in routing requests for data item 13 starting at node 1 and data item 9 starting at node 15.

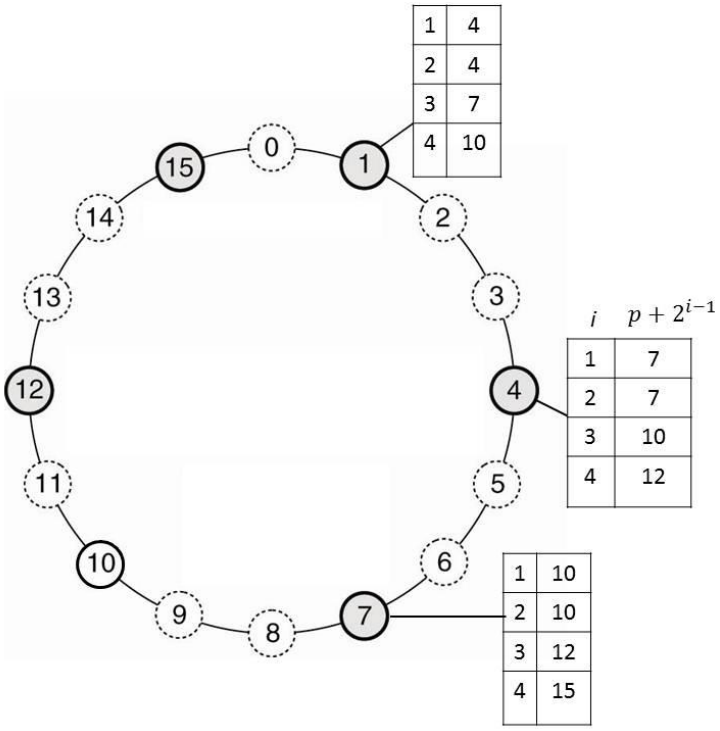

Figure Q.2

2(c) **[4 marks]**

An Overlay Network is one in which the nodes are formed by the processes and the links represent the possible communication channels. For the topology construction and management of such an architecture, Unstructured Pier to Pier (P2P) systems rely on randomized algorithms. Using a diagram illustrate and describe the 2-layered approach for such topology maintenance.

[End of Question 2]

QUESTION 3 ***[TOTAL MARKS: 25]***

Q 3(a) **[7 Marks]**

Semaphore objects in Java may be used to 'throttle' certain applications for the purposes of load balancing, for instance. In the course we covered a simple code fragment demonstrating such an application of Semaphore objects. Outline and describe this (or a similar piece of code), taking care to comment your code.

Q 3(b) **[18 Marks]**

Describe the steps involved in programming a Java RMI application. Write code that implements the Remote Database Server in Java RMI (i.e. the interface, the client

and the server). Your code should implement `read()` and `write()` methods on the database server, where `read()` returns the value of the (integer) database and `write(int value)` sets the database to be a particular (integer) value.

[End of Question 3]

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

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

In the context of concurrent programming using MPI and OpenMP *hybridization* is used to denote programs containing MPI and OpenMP code fragments. Give three ways in which combining Hybrid MPI and OpenMP code can help in optimising a Concurrent Program. Give three disadvantages of this form of programming.

Q 4(b) **[10 Marks]**

For calculating a value of $\boldsymbol{\pi}$ numerically it can be shown that:

$$\int_0^1 \frac{dx}{1+x^2} = \tan^{-1} 1 = \frac{\pi}{4}$$

Figure Q.4 shows how such an integral could be calculated numerically. Write fully commented MPI Code with C bindings to execute the calculation over **`nproc`** processors, where the calculation is performed as a discrete sum of **`num_step`** steps each of width **`step`**. In Figure Q4 this is illustrated as **`nproc=2, step=1/8`** and **`num_step=8`** but you should leave these as parameters in your code.

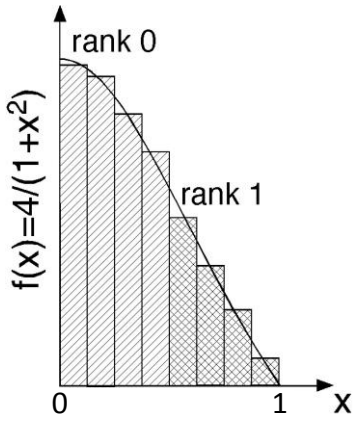

Figure Q4

Q 4(c) **[9 Marks]**

For each MPI process in Q4(b) above, **`nthreads`** OpenMP threads are to perform part of the sum. Amend your code in Q4(b) to provide OpenMP code which carries out the internal summation. Your code should use the Reduction operation internally on OpenMP nodes and a separate Reduction operation for the MPI processes.

[End of Question 4]

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

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

Describe thoroughly the three approaches to generating a SOAP Web Service in Java. If this were to be done in JAXWS what tools would be needed?

Q 5(b) **[9 Marks]**

Draw a fully annotated diagram showing the so-called "Publish-Find-Bind" model of Web Services. Describe thoroughly the role of SOAP in Web Services. List four differences between Web Services in SOAP and REST.

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

Part of a Java Interface `WeatherForecast.java` to return the weather forecast and the associated temperature of a City is shown in Figure Q5. 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 getForecast (String City){
  // implementation omitted
}

public String getTemp (String City) {
  // implementation omitted
}
```

Figure Q5 Interface `WeatherForecast.java`

[End of Question 5]

[END OF EXAM]

DUBLIN CITY UNIVERSITY

AUGUST/RESIT EXAMINATIONS 2015/2016

MODULE: CA4006 – Concurrent and Distributed Programming

PROGRAMME(S):

CASE	BSc in Computer Applications (Sft.Eng.)
ECSAO	Study Abroad (Engineering & Computing)
SHSAO	Study Abroad (Science & Health)

YEAR OF STUDY: 4,O

EXAMINERS:

Martin Crane (Ext:8974)
Dr. Ian Pitt

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.

Requirements for this paper (Please mark (X) as appropriate)

	Log Tables		*Thermodynamic Tables*
	Graph Paper		*Actuarial Tables*
	Dictionaries		*MCQ Only – Do not publish*
	Statistical Tables		*Attached Answer Sheet*
	Bible		

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

Q 1(a) **[5 Marks]**

Define Amdahl's law in words and as a formula.

Q 1(b) **[14 Marks]**

The efficiency of a parallel code is the serial cost: parallel cost ratio and is given by $E_P = T_1/(p \times T_P)$ (where T_1 is time on one processor and T_P is time on p processors).

[2 marks]

(i) Using Amdahl's law derive an expression for the efficiency E_P in terms of the number of processors *p* and the serial fraction *s*.

[2 marks]

(ii) Using the result in 1 (b) (i) above, what would be the effect on the parallel efficiency (i.e. increase or decrease) of an *increase* in the number of processors *p*? Give a concise reason for your answer.

[3 marks]

(iii) Using the result in 1 (b) (i) above, what would be the effect on the parallel efficiency (i.e. increase or decrease) of an *increase* in the size of the problem? Give a concise reason for your answer.

[3 marks]

(iv) Using the result in 1 (b) (i) above, what would be the effect on the parallel efficiency (i.e. increase or decrease) of an *increase* in the speed of the processors? Give a concise reason for your answer.

[4 marks]

(v) Derive a functional relationship which shows how the serial fraction *s* would have to vary to maintain constant efficiency.

Q 1(c) **[6 Marks]**

Suppose you are given some piece of sequential code that is to be parallelised. You are told that 20% of the code cannot be sped up at all and, of the remainder, 50% can be sped up by a factor of 3 and 50% can be sped up by a factor of 2. Using the fact that the reduced execution time, T_P, is given by the original time T_1 divided by

the speedup, S_P, derive an expression for *overall* speedup that can be achieved on the given code. Comment on your result.

[End of Question 1]

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

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

Describe fully what is the meaning and semantics of a semaphore. Write carefully commented C code which solves the Producer-Consumer Problem using circular buffers and semaphores. You may assume the existence of semaphores (`sem`) and operations on them in C as well as `produce()` and `consume()`, you may also assume buffer management operations `putItemIntoBuffer(item)`, `removeItemFromBuffer()`.

Q 2(b) **[16 Marks]**

Write down the semaphore invariants for the Producer Consumer problem in part (b). Hence, or otherwise, prove the following:

(i) There is no deadlock,
(ii) There is no starvation,
(iii) The consumer does not attempt to remove from an empty buffer.
(iv) The producer does not attempt to append to a full buffer.

[End of Question 2]

QUESTION 3 ***[TOTAL MARKS: 25]***

Q 3(a) **[3 Marks]**

What are the three major components of a Java application that uses Remote Method Invocation (RMI)?

Q 3(b) **[10 Marks]**

Interceptors are not implemented as part of the standard Remote Method Invocation package in Java. From your knowledge of how Interceptors work for a general Remote Object Invocation, outline using diagrams, an implementation of Interceptors in RMI. Your answer should include a description of what would happen at different layers and how invocations would work in the context of replicated objects.

Q 3(c) **[12 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 3. 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 3. `Factorial.java`

[End of Question 3]

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

Q 4(a) **[9 Marks]**

In the context of Distributed Mutual Exclusion, describe the following using a diagram: A simple, centralized algorithm and a distributed token ring algorithm. Compare these algorithms with respect to the number of messages which must be exchanged and the delay before entry (in terms of message times). What is the main problem with each?

Q 4(b) **[7 Marks]**

Clock Synchronization algorithms are essential for Mutual Exclusive access to shared resources in Distributed Systems. Figure Q4b shows three processes, each with its own clock, each of which runs at different rates. Explain, using a diagram, the operation of Lamport's Algorithm for synchronizing the logical clocks and ensuring partial causal ordering of the messages sent from process to process in Figure Q4b. What is the principal problem with Lamport's algorithm?

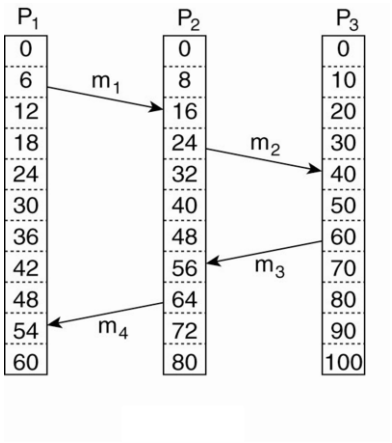

Figure Q4b.

Q 4(c) **[9 Marks]**

In many distributed algorithms, it is necessary for one process to take the role of coordinator or initiator. Dynamically choosing such a process typically happens through an election process. Figure 4c shows a distributed system comprising eight processes. The previous coordinator of the distributed system was process 7 which has now crashed. Using the example shown in Figure 4c, demonstrate the operation of the Bully Algorithm. You may assume that the processes are labelled according to their priority. Would such an algorithm scale up to large scale systems?

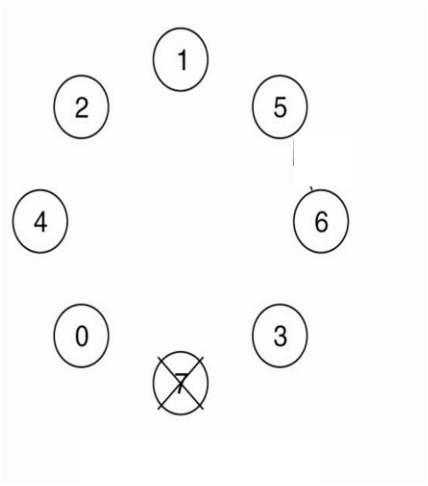

Figure 4c

[End of Question 4]

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

Q 5(a) **[8 Marks]**

In order to facilitate distribution transparency, everything in Distributed Systems is treated as an object and so clients are offered services/resources as objects that they can invoke through interfaces. Describe four such types of objects. Using a fully annotated diagram show the organization of a client-side proxy making invocations on a distributed object.

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

In the context of Web Servers, the role of a Server Cluster is important in terms of improving performance and availability. Describe fully using a diagram the operation of a *TCP handoff* in the face of increased load at the front end.

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

Part of a Java Interface `StockInterface.java` to return the Stock Market symbol and the associated Stock Value of a Company is shown below. Using Java Web Services, give the implementation of 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 getSymbol (String Company){
  // implementation omitted
}

public String getStockValue (String Company) {
  // implementation omitted
}
```

[End of Question 5]

[END OF EXAM]

DUBLIN CITY UNIVERSITY

SEMESTER 2 EXAMINATIONS 2016/2017

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)
ECSA	Study Abroad (Engineering & Computing)

YEAR OF STUDY: 4,O,X

EXAMINER(S):

Dr. Martin Crane (Ext:8974)
Prof. David Bustard
Dr. Ian Pitt

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]**

Describe Amdahl's Law for the theoretical speedup of a program executed across multiple processors. The answer should give the formula involved and explain it.

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

The efficiency of a program executed across multiple processors can be expressed as $E_P = T_1/(p \times T_P)$ (where T_1 is the time taken to execute the program on one processor and T_P, the time taken to execute the program on *p* processors).

[2 marks]

(i) Using Amdahl's law derive an expression for the efficiency E_P in terms of the number of processors *p* and the serial fraction *s*.

[3 marks]

(ii) Using the result in 1 (b) (i) above, what would be the effect on E_P (i.e. increase or decrease) of an *increase* in the size of the computational task? Explain your answer.

[3 marks]

(iii) Using the result in 1 (b) (i) above, what would be the effect on the efficiency E_P (i.e. increase or decrease) of an *increase* in the speed of the processors? Explain your answer.

Q 1(c) **[9 Marks]**

Define what is meant by Strong versus Weak Scalability. State and explain thoroughly Gustafson-Barsis' Law for *scaled speedup.* Your answer should explain using a comparative performance graph, or in words, the difference between Gustafson-Barsis' Law and Amdahl's law for speedup.

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

Vicki plans to justify her purchase of a $30 million iGadzooks supercomputer by demonstrating its 16,384 processors can achieve a scaled speedup of 15,000 on a problem of great importance to her employer. Use Gustafson-Barsis' law to find the maximum fraction of the execution time that can be devoted to inherently sequential operations if her application is to achieve this goal.

[End of Question 1]

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

Q 2(a) **[5 Marks]**

Explain the operation of a monitor and how it can be used to control access to a shared resource.

Q 2(b) **[10 Marks]**

Give a solution to the Dining Philosophers problem using monitors in C, and prove that deadlock cannot occur. Give in your answer a high level description of the algorithm.

Q 2(c) **[10 Marks]**

The Dining Schoolboys: A class of Schoolboys eats communal dinners from a large pot that can hold M servings of porridge. When a Schoolboy wants to eat, he helps himself from the pot unless it is empty in which case he waits for the pot to be filled. If the pot is empty, the cook refills the pot with M servings. The operations carried out by the cook and the Schoolboys are `fill_pot()` and `get_serving()` respectively. Model the behaviour of the pot using a monitor and implement this monitor in C. Your solution should state explicitly what the condition variables are.

[End of Question 2]

QUESTION 3 ***[TOTAL MARKS: 25]***

Q 3(a) **[9 Marks]**

Implement fully a Bounded Buffer in Java which uses ReentrantLocks. Your code should be fully commented. Explain, giving an example, what is meant by the term Reentrant Synchronization.

Q 3(b) **[16 Marks]**

The interface `PowerService.java` is shown in Figure Q3 presenting two methods, `square(int num1)` and `power(num1,num2)`, that return as `BigInteger`s, the square (of a number `num1`) and the power (of two numbers, `num1` and `num2`, such that `power=num1`$^{\texttt{num2}}$), respectively. Develop fully commented RMI code to implement the interface as follows, documenting the function of each of the following major components:

i. A class `PowerServiceServer.java` that implements the server and the methods `square(int num1)` and `power(num1,num2)`,
ii. A class `PowerServiceClient.java` that interacts with the server and calls the `square ()` and `power()` methods.

You may need to use the following methods from BigInteger Class:

i. `BigInteger(String val)`, a constructor that translates the decimal `String` (of value `val`) representation of a `BigInteger` into a `BigInteger`.
ii. `BigInteger pow(num)` a method that returns a `BigInteger` whose value is (`this`$^{\texttt{num}}$)

```
import java.math.BigInteger;
import java.rmi.*;

public interface PowerService extends java.rmi.Remote
{
      // Calculate the square of a number
      public BigInteger square ( int number )
            throws RemoteException;

      // Calculate the power of a number
      public BigInteger power  ( int num1, int num2)
            throws RemoteException;
}
```

Figure Q3: PowerService Interface

[End of Question 3]

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

Q 4(a) **[10 Marks]**

Outline two advantages and two disadvantages of both MPI and OpenMP. How might a combination of the two be advantageous in certain circumstances?

Q 4(b) **[10 Marks]**

Using the MPI_Bcast routine, implement a fully commented Matrix-Vector product on an 8 node cluster in MPI. You may assume that the Matrix is 8X8 and the vector is 8X1 and that both have been previously set up and are of type MPI_INT.

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

Using OpenMP, with number of threads set to 8 and the OpenMP reduction operation, optimise the nodal calculations in the code that you have developed in Q4(b).

[End of Question 4]

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

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

Describe with the aid of a fully labelled diagram the functionality of an Apache Web Server. What is the role of (i) the Apache Portable Runtime and (ii) the hook?

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

Part of a Java Interface `PhoneBook.java` to return the Number (as a string e.g. '35317008974') and an associated Office (e.g. 'L2.51') of a member of a department is shown Figure Q5. Using Java SOAP Web Services, give the implementation of 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 getNumber (String Name){
  // implementation omitted
}

public String getOffice (String Name) {
  // implementation omitted
}
```

Figure Q5 Interface PhoneBook.java

[End of Question 5]

[END OF EXAM]

DUBLIN CITY UNIVERSITY

AUGUST/RESIT EXAMINATIONS 2016/2017

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)
ECSA	Study Abroad (Engineering & Computing)

YEAR OF STUDY: 4,O,X

EXAMINER(S):

Dr. Martin Crane (Ext:8974)
Prof. David Bustard
Dr. Ian Pitt

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) **[8 Marks]**

Distinguish clearly between processes and threads. Show by means of a diagram and definition what is meant by Multithreading. Explain the difference between Implicit and Explicit Multithreading.

Q 1(b) **[17 Marks]**

.

[3 marks]

(i) Describe Amdahl's Law for the theoretical speedup of a program executed across multiple processors. The answer should give the formula involved and explain it.

[4 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.

[5 marks]

(iv) Gustafson-Barsis' law contrasts with Amdahl's law, as it describes a limit on the speed-up that parallelization can provide, given a fixed data set size. On the other hand, Gustafson-Barsis' law says that there is a fixed amount of parallel work per processor i.e. computations involving arbitrarily large data sets can be efficiently parallelized. Given the formula for speedup according to Gustafson-Barsis' law:

$$Speed-up, S = k(P - s(P-1))$$

where *S* is the speedup on a particular processor, *P* is the number of Processors, *s* the non-parallelizable fraction of work on any parallel process and *k* is constant, graph *S* against *P*. What are the implications of Gustafson-Barsis' law for large datasets that Supercomputers are currently applied to?

[End of Question 1]

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

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

Define Dekker's Algorithm in words and implement it in SR for two processors. Explain clearly all parts of the code.

Q 2(b) **[17 Marks]**

Peterson's algorithm is a variation on Dekker's algorithm whereby each processor uses two variables, `flag` and `turn`. A `flag` value of 1 indicates that the process wants to enter the critical section. The variable `turn` holds the process ID whose turn it is. Entrance to the critical section is granted for process P0 if P1 does not want to enter its critical section or if P1 has given priority to P0 by setting `turn` to 0.

[9 marks]

(i) Implement Peterson's algorithm in Java and briefly compare it with Dekker's algorithm.

[8 marks]

(ii) Show that, in Peterson's algorithm in (i), Mutual Exclusion is satisfied and No Starvation occurs.

[End of Question 2]

QUESTION 3 ***[TOTAL MARKS: 25]***

Q 3(a) **[8 Marks]**

What are the 4 types of processes in Message Passing Programs?

Q 3(b) **[6 Marks]**

Describe and give pseudo-code for the Synchronous Sieve of Eratosthenes algorithm with message passing. What is the principal disadvantage of the synchronous version of the algorithm?

Q 3(c) **[11 Marks]**

Implement in Java and briefly describe the Parallel Synchronous Odd-Even Exchange Sort Heartbeat Algorithm for sorting an array into ascending order over *n* processors.

[End of Question 3]

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

Q 4(a) **[3 Marks]**

Outline the steps involved in programming a Remote Method Invocation (RMI) application.

Q 4(b) **[7 Marks]**

Explain in detail the role of the RMI Registry in the context of Remote Method Invocation (RMI) applications. What is the role of the `bind` and `lookup` methods provided by the `Naming` Class? What is the role of the Security Manager in RMI applications?

Q 4(c) **[15 Marks]**

A Remote Method Invocation (RMI) Interface to allow a client to download any type of file (plain text or binary) from a remote machine is shown in Figure Q4 You are required to implement the remote interface, develop the server and develop a client that invokes the remote method `downloadFile`. Your code should be fully commented and should document the function of each major component.

```
import java.rmi.Remote;
import java.rmi.RemoteException;

public interface FileInterface extends Remote {
   public byte[] downloadFile(String fileName) throws
   RemoteException;
}
```

Figure Q4. `FileInterface.java`

[End of Question 4]

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

Q 5(a) **[7 Marks]**

Give four differences between SOAP & Rest Web services. Are there any circumstances in which it is preferable to use SOAP over Rest Web Services?

Q 5(b) **[6 Marks]**

What is meant by saying that a command is idempotent? Describe the four Rest operations, saying whether or not they are idempotent.

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

Part of an Java Interface `Addition.java` to allow a client to invoke a command to add two (long) integers is shown in Figure Q5. Using Java SOAP Web Services, give the implementation of 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 long add(long x, long y){
      // implementation omitted
}
```

Figure Q5. Interface `Addition.java`

[End of Question 5]

[END OF EXAM]

DUBLIN CITY UNIVERSITY

SEMESTER 2 EXAMINATIONS 2017/2018

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)
ECSA	Study Abroad (Engineering & Computing)

YEAR OF STUDY: 4,O,X

EXAMINER(S):

Martin Crane	(x8974)
Prof. Brendan Tangney	(External)
Dr. Hitesh Tewari	(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.

Requirements for this paper:
1. Log Tables

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

Q 1(a) **[5 Marks]**

Describe Amdahl's Law for the theoretical speedup of a program executed across multiple processors. The answer should give the formula involved and explain it.

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

Assume 0.1% of the runtime of a program is not parallelizable. This program is supposed to run on the Tianhe-2 supercomputer, which consists of 3,120,000 cores. Under the assumption that the program runs at the same speed on all of those cores, and there are no additional overheads, by Amdahl's law what is the parallel speedup on 30, 300, 30,000 and 3,000,000 cores?

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

Assume that the same program as (b) (with the same serial/parallel fraction) invokes a broadcast operation. This broadcast adds overhead, depending on the number of cores involved, *P*. There are two broadcast implementations available. One adds a parallel overhead of 0.0001 × *P*, the other one 0.0005×ln(*P*) (where ln(*P*) is the natural logarithm of *P*). For each broadcast implementation calculate the number of cores for which you get the highest speedup.

Note: $\frac{d(P)}{dP} = 1, \frac{d}{dP}\left(\frac{1}{P}\right) = -\frac{1}{P^2} and \frac{d(\log(P))}{dP} = \frac{1}{P}$

[End of Question1]

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

Q 2(a) **[6 Marks]**

Describe fully the syntax and semantics of a semaphore.

Q 2(b) **[10 Marks]**

In a drinking game, 3 drinkers is required to consume as many glasses of wholesome fermented health drink as possible over the duration of the game. Accordingly, 5 filled glasses are placed in front of the drinkers and these are replenished by 2 bartenders as necessary for the duration of the game. Write well-commented C-code solving this problem using semaphores, indicating the critical sections. You may assume the existence of the P and V semaphore operations as well as operations `fill_glass()` and `empty_glass()` as necessary.

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

For your solution in (b) show that there is (i) no deadlock (ii) no starvation and (iii) no possibility to drink from an empty glass.

[End of Question2]

QUESTION 3 ***[TOTAL MARKS: 25]***

Q 3(a) **[13 Marks]**

In the context of concurrency, what is meant by a *programming model*? How does it relate to the machine model? Explain, using diagrams, the three principal programming models in a concurrent system.

Q 3(b) **[12 Marks]**

In the context of Structured Pier to Pier (P2P) architectures, the Chord algorithm is used for managing nodes logically organized in a ring and the routing of enquiries about data items that are mapped to those entities nodes. Briefly describe the steps in the Chord algorithm. One such ring is given in Figure Q.3 below with the finger tables for nodes. Complete the finger tables for nodes 10, 12 and 15 in Figure Q.3. Give the steps in routing requests for data item 13 starting at node 1 and data item 9 starting at node 15.

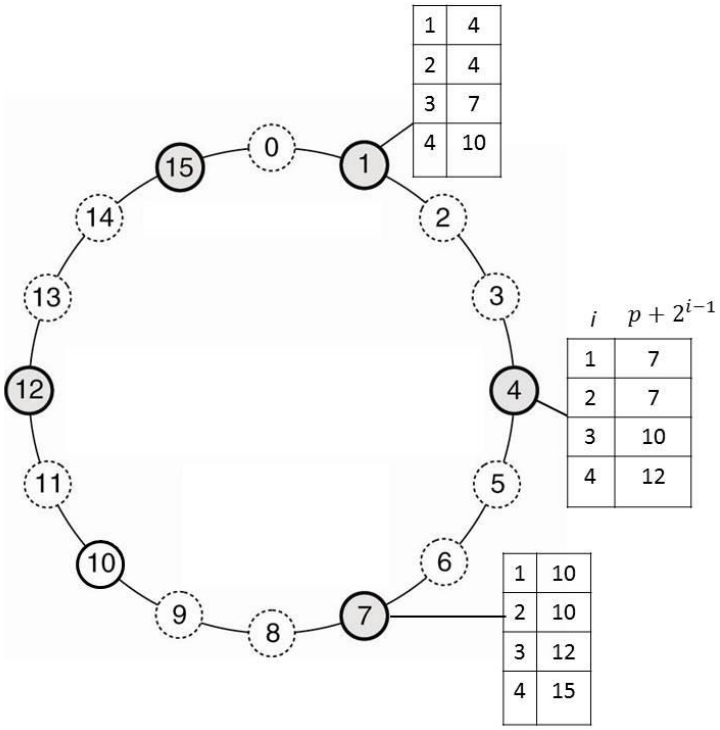

Figure Q.3

[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) **[10 Marks]**

Draw a fully annotated diagram showing the so-called “Publish-Find-Bind” model of Web Services. Describe thoroughly the role of SOAP in Web Services. Describe briefly the contents of a SOAP Message.

Q 5(b) **[15 Marks]**

Part of a Java Interface `WeatherForecast.java` to return the weather forecast and the associated temperature of a City is shown in Figure Q5. Using Java Web Services, give the implementation of 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 getForecast (String City){
  // implementation omitted
}

public String getTemp (String City) {
  // implementation omitted
}
```

Figure Q5 Interface `WeatherForecast.java`

[End of Question5]

[END OF EXAM]

DUBLIN CITY UNIVERSITY

AUGUST/RESIT EXAMINATIONS 2017/2018

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)
ECSA	Study Abroad (Engineering & Computing)

YEAR OF STUDY: 4,O,X

EXAMINER(S):

Martin Crane	(x8974)
Prof. Brendan Tangney	(External)
Dr. Hitesh Tewari	(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.

Requirements for this paper:
1. Log Tables

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

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

Briefly describe Dekker's Two-Process Mutual Exclusion Algorithm in words and implement it in C for two processors. Explain clearly all parts of the code.

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

Peterson's algorithm is a variation on Dekker's algorithm whereby each processor uses two variables, `flag` and `turn`. A `flag` value of 1 indicates that the process wants to enter the critical section. The variable `turn` holds the ID of the process whose turn it is. Entrance to the critical section is granted for process P0 if P1 does not want to enter its critical section or if P1 has given priority to P0 by setting turn to 0. Implement Peterson's algorithm in Java and briefly compare it with Dekker's algorithm.

[End of Question1]

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

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

Document fully what is meant by (i) Safety and (ii) Liveness properties.

Q 2(b) **[15 Marks]**

Briefly describe the Bakery algorithm to solve the *n* process mutual exclusion problem and implement it in C for two processors. Explain clearly all parts of the code.

Q 2(c) **[6 Marks]**

Give 2 reasons why the Bakery algorithm is not an efficient solution to the *n* process mutual exclusion problem.

[End of Question2]

QUESTION 3 ***[TOTAL MARKS: 25]***

Q 3(a) **[8 Marks]**

As part of the `java.util.concurrent` package what are `Lock` objects and how do they differ from intrinsic locks? What are `ReentrantLocks` and in what situations would one use them over `synchronized` blocks of code?

Q 3(b) **[17 Marks]**

For the code given in Figure Q3,

[7 marks]

(i) What is the meaning of the **`lock.tryLock()`** method? Explain clearly what is happening at points **A, B**. What will happen if either/both statements fail?

[3 marks]

(ii) Explain clearly what is happening at point **C**.

[7 marks]

(iii) Explain clearly what is meant by the Code at point **D**, describing briefly the possible problem that is being avoided.

```
import java.util.*;
import java.util.concurrent.*;
import java.util.concurrent.locks.*;
import static java.util.concurrent.TimeUnit.NANOSECONDS;

public class BankTransfer {
    private static Random rnd = new Random();

    public boolean transferMoney(Account fromAcct,
                                 Account toAcct,
                                 DollarAmount amount,
                                 long timeout,
                                 TimeUnit unit)
            throws InsufficientFundsException, InterruptedException {
        long fixedDelay = getFixedDelayComponentNanos(timeout, unit);
        long randMod = getRandomDelayModulusNanos(timeout, unit);
        long stopTime = System.nanoTime() + unit.toNanos(timeout);

        while (true) {
    A ->if (fromAcct.lock.tryLock()) {
            try {
        B ->if (toAcct.lock.tryLock()) {
                try {
                    if (fromAcct.getBalance().compareTo(amount) < 0)
                        throw new InsufficientFundsException();
                    else {
            C ->    fromAcct.debit(amount);
                        toAcct.credit(amount);
                        return true;
                    }
                } finally {
                    toAcct.lock.unlock();
```

```
                    }
                }
            } finally {
                fromAcct.lock.unlock();
            }
        }
        if (System.nanoTime() > stopTime)
            return false;
        NANOSECONDS.sleep(fixedDelay + rnd.nextLong() % randMod);
    }
}

private static final int DELAY_FIXED = 1;
private static final int DELAY_RANDOM = 2;

static long getFixedDelayComponentNanos(long timeout, TimeUnit unit) {
    return DELAY_FIXED;
}

static long getRandomDelayModulusNanos(long timeout, TimeUnit unit) {
    return DELAY_RANDOM;
}

static class DollarAmount implements Comparable<DollarAmount> {
    public int compareTo(DollarAmount other) {
        return 0;
    }

    DollarAmount(int dollars) {
    }
}


class Account {
    public Lock lock;

    void debit(DollarAmount d) {
    }

    void credit(DollarAmount d) {
    }

    DollarAmount getBalance() {
        return null;
    }
}

class InsufficientFundsException extends Exception {
}
                                    }
```

D

Figure Q3

[End of Question3]

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

Q 4(a) **[9 Marks]**

In the context of Distributed Mutual Exclusion, describe the following using a diagram: A simple, centralized algorithm and a distributed token ring algorithm. Compare these algorithms with respect to the number of messages which must be exchanged and the delay before entry (in terms of message times). What is the main problem with each?

Q 4(b) **[7 Marks]**

Clock Synchronization algorithms are essential for Mutual Exclusive access to shared resources in Distributed Systems. Figure Q4b shows three processes, each with its own clock, each of which runs at different rates. Explain, using a diagram, the operation of Lamport's Algorithm for synchronizing the logical clocks and ensuring partial causal ordering of the messages sent from process to process in Figure Q4b. What is the principal problem with Lamport's algorithm?

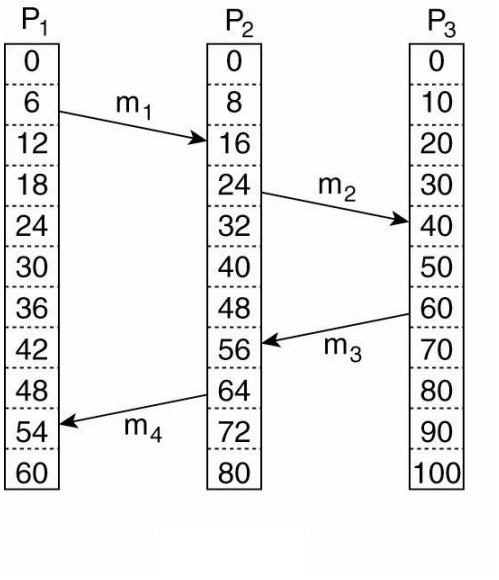

Figure Q4b.

Q 4(c) **[9 Marks]**

In many distributed algorithms, it is necessary for one process to take the role of coordinator or initiator. Dynamically choosing such a process typically happens through an election process. Figure Q4c shows a distributed system comprising

eight processes. The previous coordinator of the distributed system was process 7 which has now crashed. Using the example shown in Figure 4c, demonstrate the operation of the Bully Algorithm. You may assume that the processes are labelled according to their priority. Would such an algorithm scale up to large scale systems?

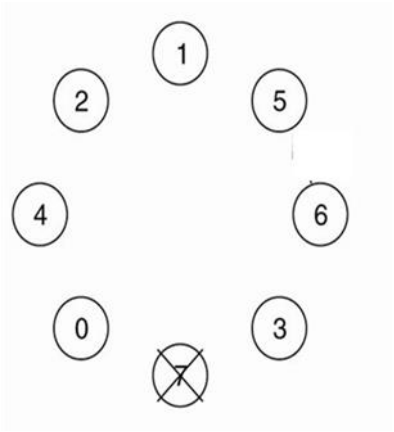

Figure Q4c

[End of Question4]

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

Q 5(a) **[10 Marks]**

Name and briefly describe the three approaches used to define Java Artifacts in SOAP web services, indicating, giving reasons, under which conditions each approach is preferable.

Q 5(b) **[15 Marks]**

Part of a Java Interface `LongTerm.java` to return the product and sum of two `long`s is shown in Figure Q5. Using Java Web Services, give the implementation of 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 long mult(in long a, in long b);
  // implementation omitted
   };

public long add(in long a, in long b);
  // implementation omitted
   };
```

Figure Q5 Interface `LongTerm.java`

[End of Question5]

[END OF EXAM]

Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

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. 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) **[12 Marks]**

Distinguish clearly between processes and threads. Differentiate, using diagrams and definitions the differences between Multithreading and Multitasking.

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

[3 marks]

(i) Define clearly, in words and as a formula, what is meant by Amdahl's law.

[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.

[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.

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()`.

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

Describe fully 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.

[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. 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.

Q 3(c) **[11 Marks]**

Write code for the sleeping barber problem using **`lock`** and **`condition`** objects in Java to implement a **`Barbershop`** class. 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.

[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.

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.

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]

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]

Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

SEMESTER 2 EXAMINATIONS 2021/2022

MODULE: CA4006 - Concurrent and Distributed Programming

PROGRAMME(S):

CASE	BSc in Computer Applications (Sft.Eng.)
ECSAO	Study Abroad (Engineering & Computing)
ECSA	Study Abroad (Engineering & Computing)

YEAR OF STUDY: 4,O,X

EXAMINER(S):

Dr. Takfarinas Saber (Internal) (Ext:6831)

TIME ALLOWED: 2 Hours

INSTRUCTIONS: **Answer all questions.**

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.

There are no additional requirements for this paper.

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

Q 1(a) **[10 Marks]**

Using an example of your choice, describe Corse grain and Fine grain task partitioning. Show advantages and disadvantages of each task partitioning.

Q 1(b) **[10 Marks]**

You are requested to design a system to handle online transactions of an E-commerce website capable of receiving and executing up to 7,200 transactions per minute at an average latency of 100ms:

- What kind of decomposition strategy would you put in place to handle that number of transactions and why?
- What is the size of the buffer that you would need to put in place in theory and in practice to achieve the performance required for the website?

Q 1(c) **[5 Marks]**

Assume that you have written a code (composed of 20% serial and 80% parallel parts) that is able to execute a single transaction for the system in Q 1(b) in 100ms on a single processor.

- What law would you use to calculate the speedup you could achieve if you execute the transaction on multiple processors? Provide its formula and calculate the speedup you would achieve with 8 processors.

[End of Question 1]

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

Q 2(a) **[5 Marks]**

What are the correctness properties that apply on programs that are not supposed to terminate? Explain the requirement for each property.

Q 2(b) **[10 Marks]**

Consider the pseudocode in Figure Q 2(b) which depicts a solution to the dining philosophers problem with 4 philosophers (each running as a separate process). Philosophers alternate between thinking and eating (after picking up the two forks beside them). In Figure Q 2(b), picking a fork `i` is protected by a lock `forks[i]`.

- Assess the program for deadlocks.
- Propose two deadlock free solutions to the problem—each addressing a different deadlock requirement. Provide a pseudocode and explain what deadlock requirement is prevented in each solution.

```
noPhilosophers = 4
forks = lock[noPhilosophers]

philosopher(i) {
|     while(1) {
|     |     think()
|     |     forks[i].acquire()
|     |     forks[(i+1)%noPhilosophers].acquire()
|     |     eat()
|     |     forks[(i+1)%noPhilosophers].release()
|     |     forks[i].release()
|     }
}
main() {
|     for(i=0; i<noPhilosophers; i++) {
|     |     philosopher(i)
|     }
}
```

Figure Q 2(b). Pseudocode Solution for Dining Philosophers Problem

Q 2(c) **[10 Marks]**

Analyse the java code provided in Figure Q 2(c). What would be the outcome of the method call `calculateAndPrint(3)`? Explain your answer. What would you change in the code to make it deterministic while maintaining concurrency? Provide the corresponding code with comments.

```
public class FactorialComputation{
|     private static boolean completed=false;
|     private static double result=0.0;
|
|     private class ResultPrinter extends Thread{
|     |     public void run(){
|     |     |     while (!completed)
|     |     |     |     Thread.yield();
|     |     |     System.out.println(result);
|     |     }
|     }
|     public double factorial(int x){
|     |     double fact = 1.0;
|     |     for (int i=2; i<=x; i++){ fact = fact * i;}
|     |     return fact;
|     }
|     public void calculateAndPrint(int number){
|     |     ResultPrinter  rp = new ResultPrinter();
|     |     rp.start();
|     |     result = factorial(number);
|     |     completed = true;
|     }
}
```

Figure Q 2(c). Java code for factorial computation

[End of Question 2]

QUESTION 3 ***[TOTAL MARKS: 25]***

Q 3(a) **[9 Marks]**

Various Pi (π) approximations have been proposed over time. One of them is the *Leibniz formula* which states that:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots$$

This equation could be re-written as:

$$\frac{\pi}{4} = \frac{1}{1 + 2*0} - \frac{1}{1 + 2*1} + \frac{1}{2*2} - \frac{1}{1 + 2*3} + \frac{1}{1 + 2*4} - \cdots +/- \frac{1}{1 + 2*n}$$

Write a fully commented C code to compute *Leibniz approximation* of Pi to a precision `n` and using `nThreads` OpenMP threads.

Q 3(b) **[10 Marks]**

Amend your code for Q 3(a) to enable the computation of *Leibniz approximation of Pi to a precision* `n` over `nProc` MPI processors, with `nThread` OpenMP threads to perform part of the MPI processors internal calculation.

Q 3(c) **[6 Marks]**

What program decomposition strategy have you used in Q3(b) and what patterns have you put in place to implement it? What factors have you considered for adopting your decomposition strategy?

[End of Question 3]

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

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

You are requested by a credit card provider to build a distributed system to handle credit card payments of highway tolls in Europe. Explain what design aspects would you consider and what design aspect would you prioritise?

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

With the help of diagrams and/or code snippets, describe a design you would put in place for the distributed system described in Q 4(a). Indicate the rational for making each design choice. What are implications of your design choices on your system?

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

Assume that you have implemented the Remote Procedure Call concept to send transactions between a toll payment terminal and a server. What would be the limitations of such implementation?

[End of Question 4]

[END OF EXAM]

Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

AUGUST/RESIT EXAMINATIONS 2021/2022

MODULE: CA4006 - Concurrent and Distributed Programming

PROGRAMME(S):

CASE	BSc in Computer Applications (Sft.Eng.)
ECSAO	Study Abroad (Engineering & Computing)
ECSA	Study Abroad (Engineering & Computing)

YEAR OF STUDY: 4,O,X

EXAMINER(S):

Dr. Takfarinas Saber	(Internal)	(Ext:6831)

TIME ALLOWED: 2 Hours

INSTRUCTIONS: **Answer all questions.**

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.

There are no additional requirements for this paper.

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

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

With the help of an example, define latency and throughput. Describe two systems: one where the goal is to minimise latency and another one where the goal is to maximise throughput. What is the computing environment associated with each goal?

Q 1(b) **[10 Marks]**

You have designed a system that is composed of 8 components. Figure Q 1(b) shows the organisation of the components and their respective throughputs (in request per second).

- What is the overall throughput that the system could achieve? Explain your answer.
- You were told that the bottleneck component is run on the cloud and the cloud provider charges $1,000 for every 1 request increase per second in throughput. To improve the performance of the system, your company pays the cloud provider to increase the throughput to 120 Req/s. Is that a good idea and why?

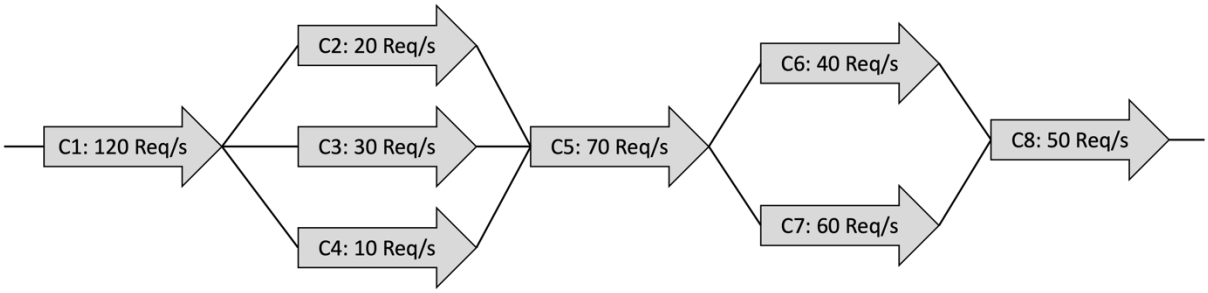

Figure Q 1(b). Throughput of each component composing the system

Q 1(c) **[7 Marks]**

You have written a program that takes 100ms on a single processor. If you were told that you could achieve a speedup of 2 on 4 processors, compute the percentage of serial and parallelisable sections in your program. How much speedup would you achieve on 8 processors?

[End of Question 1]

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

Q 2(a) **[10 Marks]**

What are the advantages of monitors over spinlocks? What is the mechanism that monitor uses to achieve each advantage? Show two alternatives for using monitor in Java?

Q 2(b) **[5 Marks]**

One of the monitor concepts is the monitor invariant. Describe this concept. In which case the monitor invariant is relaxed? Explain with the help of an example.

Q 2(c) **[10 Marks]**

Analyse the code in Figure Q 2(c) which is supposed to manage operations on a bank account and print its balance (but only when the balance is negative). Describe in words the working of the code in Figure Q 2(c). Describe an example where the code would not work as expected. Describe a solution you would bring to solve the problem. Provide a code snippet that shows your solution.

```
Class Account{
      private double balance = 0.0;
      public void synchronized deposit(double d){
            int balance = balance + d;
      }
      public void synchronized withdraw(double w){
            int balance = balance - w;
            If (balance < 0)
                  notify();
      }
      public void synchronized printBalance(){
            if (balance >= 0)
                  wait();
            System.out.println("Balance=" + balance);
      }
}
```

Figure Q 2(c). Java Code for Bank Account

[End of Question 2]

QUESTION 3 ***[TOTAL MARKS: 25]***

Q 3(a) **[10 Marks]**

Explain the general idea behind the 'Epidemic' algorithm in Replication Systems for Distributed Systems? Explain its relationship with epidemics (rapid spread of infectious disease) in a community.
What assumptions should a Replication System meet to enable the implementation of the epidemic algorithm.
Using an example, describe two epidemic propagation models. Compare them in terms of complexity and reliability.

Q 3(b) **[8 Marks]**

One approach to ensure mutual exclusion in distributed systems is using the centralised algorithm. Describe this algorithms on an example. What are characteristics of this algorithm in terms of number of exchanged messages and delay before entry.

Q 3(c) **[7 Marks]**

Before implementing the centralised mutual exclusion in a distributed system, it is necessary to dynamically elect one node to take the role of coordinator. Describe an algorithm that will allow you to achieve that. Show the working of this algorithm on a peer-to-peer distributed system with 8 nodes.

[End of Question 3]

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

Q 4(a) **[8 Marks]**

Using an example, explain the concepts of 'location transparency', 'location independence', and 'cache' in the context of a Distributed File System? What are the benefits of using a cache for such system?
What is referred to the 'cache-consistency' problem? How can you solve it?

Q 4(b) **[10 Marks]**

Three hosts are communicating with one another in a distributed system. Their clocks run at different speeds:

- Process A has dC/dt = 1.2 (a fast clock)
- Process B has dC/dt = 1 (a perfect clock)
- Process C has dC/dt = 0.8 (a slow clock)

The following messages are sent (T denotes real time in seconds).

- A sends a message to B at time T=20
- A sends a message to C at time T=50
- A sends a message to B at time T=60
- C sends a message to A at time T=80
- B sends a message to C at time T=90

Assuming all messages take exactly 10 real-time seconds to arrive:
What are the timestamps of the three processes if we use Lamport's algorithm to enforce a global logical clock? Explain each step of your answer.

Q 4(c) **[7 Marks]**

What is the main issue with Lamport's algorithm when enforcing a global logical clock? Provide an example showcasing the problem. Briefly name and describe an alternative algorithm that would solve the problem faced by Lamport's algorithm.

[End of Question 4]

[END OF EXAM]

Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

SEMESTER 2 CRA EXAM 2019/2020

MODULE: CA4006 - Concurrent and Distributed Programming

PROGRAMME(S):

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

YEAR OF STUDY: 4,O

EXAMINER(S):

Dr. Rob Brennan	(Internal)	6008

TIME ALLOWED: 3 Hours and 30 minutes

INSTRUCTIONS: Answer 4 questions from the 5 provided. All questions carry equal marks.

Please type all code and pseudo-code rather than using images/screenshots.

This is an open book exam so it is expected that wherever possible you state things in your own words (i.e. do not just cut and paste). Please reference all your sources of images or quotes (supply a URL or academic reference).

Please submit by the “due date” in Loop. There is up to an extra 1 hour for people who have technical difficulties, in which case you must contact DESC (DCU Examination Support Centre):
examsupport@dcu.ie or 01 700 6151

Rob is online for clarifications during the exam: rob.brennan@dcu.ie but all serious issues should go to DESC.

Please monitor your email during the exam in case we need to make an announcement.

OTHER RESTRICTIONS.
All answers will automatically be checked for similarity with internet sources and other students. Exams submitted late for technical difficulties without contacting DESC may have marks deducted. 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.

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

Q 1(a) **[16 Marks]**

Define and distinguish clearly between HPC and HTP systems and give an example of a typical application for each type of system. Explain the relative importance of latency for these systems, e.g. using Gustafson-Barsis' Law. Hence briefly describe the types of hardware architectures each system is likely to employ and explain why they are the most suitable.

Q 1(b) **[5 Marks]**

How many tasks must be buffered in a system if it receives 10,000 transactions per minute and the average task latency is 100ms?

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

How would you decompose the work done by the transaction processing system described in Q1(b) to support concurrency? Explain why you picked this strategy.

[End of Question 1]

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

Q 2(a) **[10 Marks]**

What are the requirements for deadlock to occur in concurrent code? Which requirement has been the most important in your studies or projects to date? Explain, using an example, why this is the case.

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

With reference to the Concurrent Code below, answer the questions below:

```
01 public class Account {
02    private AtomicInteger balance = new AtomicInteger(0);
03
04    public int withdraw (int amount){
05          int balance = this.balance.get();
06          balance = balance – amount;
07          this.balance.set(balance};
08          return balance;
09    }
10 }
```

Figure Q2. Bank Account

i) Where is the critical section in this code?
ii) Does the AtomicInteger on line 02 protect the account balance effectively? Explain your answer.
iii) Is this code threadsafe? Explain your answer.
iv) Re-write the code to be more efficient and concurrently correct.

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

Modify the code in fig Q2 above to give a correct concurrent transaction class capable of transferring money from one account to another by checking the available balance first and ensuring that no funds are lost mid-transaction due to the activity of many threads in the system.

[End of Question 2]

QUESTION 3 ***[TOTAL MARKS: 25]***

Q 3(a) **[8 Marks]**

Explain the operation of a monitor and how it can be used to control access to a shared resource. Include references to the diagram in fig. Q3 below. Describe a scenario, e.g. using pseudo-code, where the threads A, B and C manipulate the shared data in that order due to the operation of the monitor.

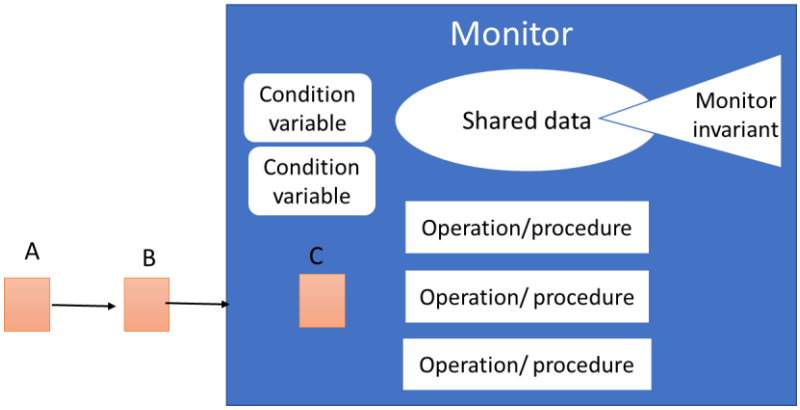

Figure Q3: A Monitor

Q 3(b) **[6 Marks]**

Describe at least three features (or pitfalls or limitations) of `synchronized` keyword monitors in Java. What is the most common problem you encountered with the `synchronized` keyword in your project work? (Explain your reasons, with an example).

Q 3(c) **[11 Marks]**

Write code for the sleeping barber problem using `lock` and `condition` objects in java to implement a `Barbershop` class. 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.

[End of Question 3]

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

Q 4(a) **[10 Marks]**

What parallel processing design pattern is illustrated in figure Q4 below? Describe the main features of this pattern. In figure Q4 what would be the optimum number of processors for the tasks illustrated? Explain the rationale for your answer. Describe an example use case for this pattern explaining why it is especially applicable in your opinion.

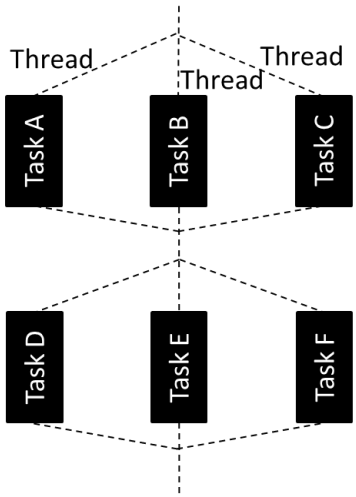

Figure Q4: Parallel Pattern

Q 4(b) **[10 Marks]**

Explain with the aid of a worked example the operation of the following distributed MPI code for calculation of Pi. Describe each MPI command used and why it is important for parallel execution of the program.

[continued on next page]

```
int main(int argc, char *argv[]) {
MPI_Init(&argc, &argv);
#define INT_MAX_ 1000000000
int myid, size, inside=0, outside=0, points=10000;
double x,y, Pi_comp, Pi_real=3.14159265358979323846264 3;
MPI_Comm_rank(MPI_COMM_WORLD, &myid);
MPI_Comm_size(MPI_COMM_WORLD, &size);
MPI_Bcast(&points,1,MPI_INT, 0, MPI_COMM_WORLD);
rands=new double[2*points];
for (int i=0; i<2*points; i++ ){
      rands[i]=random();
      }
for (int i=0; i<points;i++ ){
      x=rands[2*i]/INT_MAX_;
      y=rands[2*i+1]/INT_MAX_;
      if((x*x+y*y)<1) inside++      /* point is inside unit circle so incr
var inside */
}
delete[] rands;
if (myid == 0) {
      for (int i=1; i<size; i++) {
            int temp;
            MPI_Recv(&temp, 1, MPI_INT, i, i, MPI_COMM_WORLD,
                MPI_STATUS_IGNORE);
            inside+=temp;     }
}
else
      MPI_Send(&inside, 1, MPI_INT, 0, i, MPI_COMM_WORLD);

MPI_Reduce(&inside,&total,1,MPI_INT,MPI_SUM,0, MPI_COMM_WORLD);
if (myid == 0) {
      Pi_comp = 4 * (double) inside / (double)(size*points);
      cout << "Value obtained: " << Pi_comp << endl << "Pi:" << Pi_real <<
endl;}
MPI_Finalize(); return 0;
}
```

Figure 4 MPI Code for Calculation of Pi

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

How could the code in part (b) be made to scale more efficiently for compute-intensive problems with high communication costs and extreme locality? Explain your assumptions.

[End of Question 4]

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

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

Give an example of a commercial distributed system. Describe two standard types of system design choices for distributed systems in terms of: the choices available, complications/issues of distributed systems, and design problems to be addressed. Give an example of a distributed design choice you made in your project work and explain what worked (or did not) and why.

Solutions of Question5 part a

Example commercial distributed system [1 mark]

The types of design choices we must make are based on i) architecture, ii) implementation, iii) performance, iv) fault tolerance and v) consistency.

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

For the RPC system illustrated in the UML activity diagram in fig Q5 below:

(i) Describe what is happening in the diagram.
(ii) What type of delivery semantics is illustrated? Explain why.
(iii) Write carefully commented pseudo code for the RPC client and server to support this semantics.

[continued on next page]

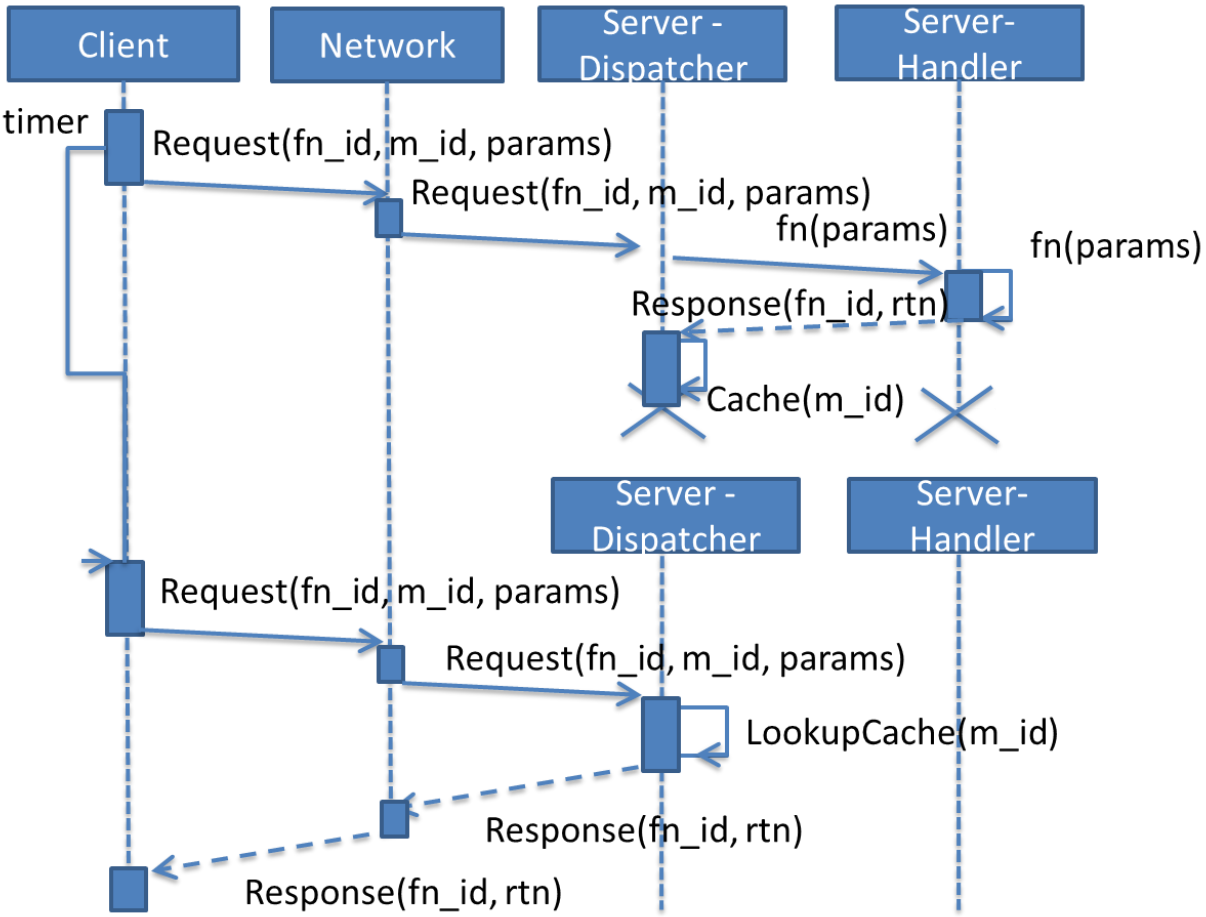

Figure Q5 An Asynchronous RPC System

[End of Question 5]

[END OF EXAM]