Time permitted: 3 hours + 10 minutes reading time

Answer all questions.

Total Marks = 85

- Answer Part A in pencil on the provided multiple choice answer sheet.
- Parts B, C and D must be answered in separate answer booklets.
- Label each booklet with a B, C or D, as well as your name and student ID.
- At end of exam, place booklets C and D inside booklet B.

This paper may not be retained by the candidate.

No examination materials are permitted.

Answers in booklets must be written in ink.
Part A. Multiple Choice Questions [25 marks]
Select just one best choice. There is no penalty for wrong answers.

1. Dynamic binding …
   A. …
   B. …
   C. …
   D. …

2..5
Java & Scala Collections:
- Recognise use and implementation idioms for common Design Patterns
  - (Adaptor, Decorator, Iterator, Factory Method)
Java & Scala generics
- Use of type parameters for methods and classes
- Type bounds
- Wild cards
- Type erasure and raw types

8..15
Concurrency control in Java
- Volatiles and atomic variables.
- Use of wait, notify, notifyAll. Nested monitors and deadlock
- Read-write locks. Reader or writer preference.
- Role of copy-on-write.

6,7, 16 .. 25
Scala
- Functions as objects
- Class abstractions in Scala (trait, class, case class, object). Their different roles
- Use of companion objects
- Type members, abstract and concrete
- Pattern matching
- Operator expressions as method calls
- Immutability
- Covariance and contravariance
- For loop syntax. Type of for expressions using yield.

e.g. Which one of the following is false? For the following method definition in Scala ...

A. the call … yields a run-time error
B. the result type of $f$ is …
C. the call … returns 4
D. the argument type of $f$ is …

Part B. Scala programming [20 marks]
Start a NEW answer booklet. Label it B on front.

1. [15 marks] You are required to implement a simple interpreter for … in Scala, which you should do in the following steps.

   a. Write down a simple grammar (using extended BNF notation) for a language of … that include the following forms

      ...

      [More description of language here]

   b. Using the Scala parser combinators, and assuming that
      
      variable : Parser [String]
      number   : Parser [Double]
      
      are given, write a parser for recognizing expressions defined by your grammar above.

   c. Extend your recognizer to a parser for evaluating expressions
      
      exp : Parser [Double]
      
      [More description of language here]

2. [5 marks] Given Scala function declarations

   ...

   a. Define ... using a [state form of recursive alogorithm] so that it satisfies:

      [More description here]

      .

   b. Discuss how you could parallellize this … in Scala. You do not need to provide full code details, but do provide enough detail to make the key issues clear.
**Part C. Java Concurrency** [20 marks]

*Start a NEW answer booklet. Label it C on front.*

A one page description of a concurrency safety problem is given.

In this question, you are required to write Java code for a number of classes, making use of the code below.

...  

Write Java code according to the following requirements. In (a) and (b), you only need to ensure the safety properties of the problem. In (c), you need to modify your code to prevent starvation.

You must not use any concurrency library utilities. Instead you must only use Java concurrency primitives, such as `synchronized`, `wait` and `notifyAll`.

(a) Write a … class. It must preserve the … safety property, i.e. …

(b) Write the code for … some repeated behaviour which must also be safe.

(c) Description of a fairness requirement that must be implemented as well, using concurrency control. It must preserve the … safety property, as well as ensure that … get the opportunity to … [enter a critical section]
Part D. Short answer questions [20 marks]
Start a NEW answer booklet. Label it D on front.

7 questions. 3 marks each except for one worth 2 marks.

Topics covering both Scala and Java, selected from:

- OO vs FP
- Static methods vs dynamic binding
- Mixins in Scala
- Abstract type members
- Structural types in Scala (read up about this!!!)
- Double-checked locking
- Happens-before relation in Java
- Single vs multiple inheritance in OO
- Role of implicits in Scala
- Basic use of ForkJoin framework

END OF PAPER