COMPUTER SCIENCE TRIPOS Part IA

NATURAL SCIENCES TRIPOS Part IA (Paper CS/1)

POLITICS, PSYCHOLOGY, AND SOCIOLOGY TRIPOS Part I (Paper 9)

Monday 4 June 2012 1.30 to 4.30

COMPUTER SCIENCE Paper 1

Answer ***five*** *questions.*

At least ***one*** *question from each section is to be answered.*

Submit the answers in five ***separate*** *bundles, each with its own cover sheet. On each cover sheet, write the numbers of* ***all*** *attempted questions, and circle the number of the question attached.*

You may not start to read the questions printed on the subsequent pages of this question paper until instructed that you may do so by the Invigilator

STATIONERY REQUIREMENTS
Script paper
Blue cover sheets
Tags

SPECIAL REQUIREMENTS
Approved calculator permitted

SECTION A

1 Foundations of Computer Science

Recall that a dictionary of (*key*, *value*) pairs can be represented by a binary search tree. Define the *union* of two binary search trees to be any binary search tree consisting of every node of the given trees.

(*a*) Write an ML function `union` to return the union of two given binary search trees. [*Note:* You may assume that they have no keys in common.] [6 marks]

Define a *slice* of a binary search tree to be a binary search tree containing every (*key*, *value*) node from the original tree such that $x \leq \mathit{key} \leq y$, where x and y are the given endpoints.

(*b*) Write an ML function `takeSlice` to return a slice – specified by a given pair of endpoints – from a binary search tree. [4 marks]

(*c*) Write an ML function `dropSlice` to *remove* a slice from a binary search tree: given a tree and a pair of endpoints, it should return the binary search tree consisting of precisely the nodes such that $x > \mathit{key}$ or $\mathit{key} > y$. [*Hint:* First consider the simpler task of deleting a node from a binary search tree.] [8 marks]

(*d*) The tree t need not be identical to that returned by

```
union(takeSlice(t, x, y),
      dropSlice(t, x, y))
```

Briefly explain how such an outcome is possible. [2 marks]

[*Note:* All ML code must be explained clearly and should be free of needless complexity.]

2 Foundations of Computer Science

(*a*) Write brief notes on `fn`-notation and curried functions in Standard ML. Illustrate your answer by presenting the code for a polymorphic curried function `replicate`, which given a non-negative integer n and a value x, returns the list $[\underbrace{x, \ldots, x}_{n}]$. [6 marks]

(*b*) Write brief notes on references in Standard ML. Illustrate your answer by discussing (with the aid of a diagram) the effect of the following two top-level declarations:

```
val rlist = (replicate 4 (ref 0)) @ (map ref [1, 2, 3, 4]);
val slist = map (fn r => ref (!r)) rlist;
```

[6 marks]

(*c*) The following three lines are typed at the ML top level, one after the other. What value is returned in each case? Justify your answer clearly. [*Note:* Recall that an expression of the form v `:=` E has type `unit`.]

```
map (fn r => (r := !r + 1)) rlist;
map (fn r => (r := !r - 1; !r)) rlist;
map (fn r => (r := !r +3; !r)) slist;
```

[8 marks]

SECTION B

3 Discrete Mathematics I

(*a*) Which of the following formulas are tautologies? Explain what is meant by "tautology" and write down truth tables to justify your answers.

(*i*) $\mathrm{p} \Rightarrow \mathrm{q}$

(*ii*) $(\mathrm{p} \Rightarrow \mathrm{q}) \Rightarrow \mathrm{p}$

(*iii*) $((\mathrm{p} \Rightarrow \mathrm{q}) \Rightarrow \mathrm{p}) \Rightarrow \mathrm{p}$ [4 marks]

(*b*) Recall the following introduction and elimination rules for implication.

$\ldots$

m. Assume P
$\ldots$
n. Q from $\ldots$ by $\ldots$

$n+1$. $P \Rightarrow Q$ from m–n,
by $\Rightarrow$-introduction.

$\ldots$
l. $P \Rightarrow Q$ from $\ldots$ by $\ldots$
$\ldots$
m. P from $\ldots$ by $\ldots$
$\ldots$
n. Q from l, m
by $\Rightarrow$-elimination.

(*i*) Write down the elimination rules for negation and falsity. [3 marks]

(*ii*) Using the four rules above, write down a structured proof of

$$\neg\mathrm{p} \Rightarrow (\mathrm{p} \Rightarrow \mathrm{q})$$

[4 marks]

(*iii*) Write down the principle of proof by contradiction. [2 marks]

(*iv*) Using everything from part (*b*) so far, write down a structured proof of

$$((\mathrm{p} \Rightarrow \mathrm{q}) \Rightarrow \mathrm{p}) \Rightarrow \mathrm{p}$$

[7 marks]

4 Discrete Mathematics I

(*a*) Let A be the set $\{1, 2, 3\}$. The following relations are subsets of $A \times A$. Draw them as directed graphs.

(*i*) $R_1 = \{(x, y) \mid x \in A \wedge y \in A \wedge x - y = 1\}$

(*ii*) $R_2 = \{(x, y) \mid x \in A \wedge y \in A \wedge x - y \geq 1\}$

(*iii*) $R_3 = \{(x, y) \mid x \in A \wedge y \in A \wedge x - y = 0\}$

(*iv*) $R_4 = \{(x, y) \mid x \in A \wedge y \in A \wedge \neg(x - y = 0)\}$

(*v*) $R_5 = \{(x, y) \mid x \in A \wedge y \in A \wedge \forall u. \exists v. x + u = y + v\}$
where u and v range over the integers

(*vi*) $R_6 = \{(x, y) \mid x \in A \wedge y \in A \wedge \exists u. \forall v. x + u = y + v\}$
where u and v range over the integers [6 marks]

(*b*) Write down what it means for a relation to be transitive. Which of the relations in part (*a*) are transitive? [3 marks]

(*c*) Write down the introduction and elimination rules for the universal quantifier in structured proof. [3 marks]

(*d*) Recall the following introduction and elimination rules for implication.

$\ldots$

$\boxed{\begin{array}{l} m.\ \text{Assume } P \\ \ldots \\ n.\ Q \text{ from } \ldots \text{ by } \ldots \end{array}}$

$n+1.\ P \Rightarrow Q$ from m–n,
by $\Rightarrow$-introduction.

$\ldots$

$l.\ P \Rightarrow Q$ from $\ldots$ by $\ldots$

$\ldots$

$m.\ P$ from $\ldots$ by $\ldots$

$\ldots$

$n.\ Q$ from l, m
by $\Rightarrow$-elimination.

Write down a structured proof of the following statement.

$$\big(\forall a. P(a) \Rightarrow Q(a)\big) \Rightarrow \Big(\big(\forall b. Q(b) \Rightarrow R(b)\big) \Rightarrow \big(\forall c. P(c) \Rightarrow R(c)\big)\Big)$$

[8 marks]

SECTION C

5 Algorithms I

(*a*) Explain the quicksort algorithm. [*Note:* Pseudocode allowed but not required. Clarity of ideas, conciseness and legibility definitely required, but not absolute completeness.] [3 marks]

(*b*) Assume that, under certain hypothetical circumstances, quicksort always partitions into two regions of relative size α and $(1-\alpha)$, with α a constant in the range $0 < \alpha < 0.5$. Under those circumstances, and ignoring rounding issues, derive an approximate expression for the minimum depth of a leaf in the recursion tree as a function of n and α. Clearly explain your derivation. [5 marks]

(*c*) How long will quicksort take if all the elements are equal? Clearly explain your derivation. [6 marks]

(*d*) It has been suggested that the pivot should be selected at random. What are the advantages and disadvantages of this strategy? How will it affect the worst-case and average-case asymptotic complexity? Discuss. [6 marks]

6 Algorithms I

(*a*) Imagine that the search procedure that looks for a key in a binary search tree is instrumented to print out the sequence of the keys of the nodes it visits.

(*i*) For each of the following sequences, say whether or not it could have been printed by that procedure, justifying any negative answers.

A) 903, 478, 551, 598, 560, 557, 555.
B) 825, 302, 811, 340, 812, 345, 363.
C) 788, 359, 875, 283, 118, 941, 466.

[3 marks]

(*ii*) Give a clear and simple description of a linear algorithm that, given an arbitrary sequence of integers, says whether or not it could have been printed by the search procedure referred to above. [*Note:* Pseudocode optional, clarity necessary] [5 marks]

(*b*) Compare the binary search tree and the binary min-heap. [*Note:* For simplicity ignore the payloads, assume that keys are integers and assume that there are no duplicate keys.]

(*i*) Give a necessary and sufficient criterion to decide whether a given binary tree is a binary search tree. [3 marks]

(*ii*) Give a necessary and sufficient criterion to decide whether a given binary tree is a binary min-heap. [3 marks]

(*iii*) Choose either the binary search tree or the binary min-heap, then give clear and concise pseudocode to output the keys of that type of tree in sorted order in linear time. Justify why your answer gives the intended results.

[3 marks]

(*iv*) For the other type of tree not chosen in part (*b*)(*iii*), is it also possible to output the sorted keys in linear time? Justify your answer. [3 marks]

SECTION D

7 Floating-Point Computation

(*a*) Briefly describe the steps used in floating-point multiplication. [3 marks]

(*b*) What exceptions are possible during floating-point multiplication and at what stage in the process of part (*a*) should they be checked? [3 marks]

(*c*) Give code or pseudocode that divides a binary-coded floating-point variable by the constant number ten (1010 in binary). [*Note:* Full marks awarded for code or pseudocode optimised for this constant divisor.] [7 marks]

(*d*) Explain the principle of the following iteration:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

[2 marks]

(*e*) A programmer writes the following iteration for the square-root of A.

$$x_{n+1} = \frac{\frac{A}{x_n} + x_n}{2}$$

Show that this is a Newton–Raphson iteration (or otherwise explain its basis) and say approximately how many steps it should take to converge for single and double-precision floating point. [5 marks]

8 Object-Oriented Programming with Java

(*a*) Write an immutable Java class `Complex` that represents a complex number with integer real and imaginary components. [*Note:* Your class should contain only methods that are essential for its use. Do not incorporate any mathematical operations in the class.] [4 marks]

(*b*) Without using Generics, adapt the class to support the use of arbitrary types to store the real and imaginary components. Give three disadvantages of this approach and comment on the immutability of the new class. [7 marks]

(*c*) Rewrite your class from part (*b*) using Generics. Discuss the extent to which this new version addresses the problems identified in part (*b*). [7 marks]

(*d*) Explain carefully why the following code cannot be used to print out an object of type `LinkedList<Complex<Double>>`. You may assume the existence of a working `print` method within each `Complex` object.

```
void printAll(LinkedList<Complex<Object>> list) {
   for (Complex<Object> c : list)
      c.print();
}
```

[2 marks]

END OF PAPER