COMPUTER SCIENCE TRIPOS Part IA

NATURAL SCIENCES TRIPOS Part IA (Paper CS/1)

POLITICS, PSYCHOLOGY, AND SOCIOLOGY TRIPOS Part I (Paper 7)

Monday 1 June 2009 1.30 to 4.30

COMPUTER SCIENCE Paper 1

*Answer **one** question from each of Sections A, B and C, and **two** questions from Section D.*

*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

(*a*) The polymorphic curried function `delFirst` takes two arguments, a predicate (Boolean-valued function) `p` and a list `xs`. It returns a list identical to `xs` except that the first element satisfying `p` is omitted; if no such element exists, then it raises an exception. Code this function in ML. [4 marks]

(*b*) Use the function `delFirst` to express the polymorphic function `delFirstElt`, where `delFirstElt x xs` returns a list identical to `xs` except that it omits the first occurrence of `x`. [2 marks]

(*c*) Carefully explain the polymorphic types of these two functions, paying particular attention to currying and equality. [4 marks]

(*d*) A list `ys` is a *permutation* of another list `xs` if `ys` is obtained by rearranging the elements of `xs`. For example, [2,1,2,1] is a permutation of [2,2,1,1]. Code an ML function to determine whether one list is a permutation of another. [4 marks]

(*e*) A list `ys` is a *generalised permutation* of `xs` if `ys` is obtained by rearranging the elements of `xs`, where one element of `xs` is specially treated: it may appear any number of times (including zero) in `ys`. For example, [1,2,1] is a generalised permutation of [1,2] but [1,2,2,1] is not because two elements (1 and 2) appear the wrong number of times in it. Code an ML function to determine whether one list is a generalised permutation of another. [6 marks]

All ML code must be explained clearly.

2 Foundations of Computer Science

(*a*) Write brief notes on top–down merge sort, contrasting it with insertion sort. State its worst-case and average-case complexity, with brief justification. (There is no need to present ML code.) [5 marks]

(*b*) Write brief notes on preorder, inorder and postorder tree traversal. Present efficient code for one of them and state, with justification, its worst-case complexity. [5 marks]

(*c*) The binary search tree t_1 is *superseded by* t_2 provided every (key, value) entry in t_1 is also present in t_2. Code an ML function to determine whether one binary search tree is superseded by another. Express its cost in terms of n_1 and n_2, the numbers of entries in t_1 and t_2, respectively. For full credit, the worst-case cost should be no worse than $O(n_1 + n_2)$. [10 marks]

All code must be explained clearly. You may assume that any necessary ML data structures or functions are available.

SECTION B

3 Discrete Mathematics I

(*a*) State the structured-proof rules for implication introduction and disjunction elimination. [3 marks]

(*b*) Give either a structured proof or a counterexample for each of the following.

(*i*) $((P \Rightarrow Q) \vee (P \Rightarrow R)) \Rightarrow (P \Rightarrow (Q \vee R))$

(*ii*) $((P \wedge Q) \Rightarrow R) \Rightarrow ((P \Rightarrow R) \wedge (Q \Rightarrow R))$

[8 marks]

For a set of sets A, write $\bigcup A$ for the set $\{x \mid \exists X \in A.x \in X\}$. For a non-empty set of sets A, write $\bigcap A$ for the set $\{x \mid \forall X \in A.x \in X\}$.

(*c*) Suppose $A \subseteq \mathcal{P}(X)$ and $B \subseteq \mathcal{P}(X)$. Prove or give a counterexample for each of the following.

(*i*) If $\bigcup A$ and $\bigcup B$ are disjoint, then A and B are disjoint.

(*ii*) If A and B are disjoint then $\bigcup A$ and $\bigcup B$ are disjoint.

(*iii*) If A and B are non-empty and $\forall X \in A.\forall Y \in B.X \subseteq Y$ then $\bigcup A \subseteq \bigcap B$.

[9 marks]

4 Discrete Mathematics I

(*a*) Define what it means for a relation $R \subseteq A \times A$ to be:

(*i*) irreflexive [1 mark]

(*ii*) symmetric [1 mark]

(*iii*) antisymmetric [1 mark]

(*b*) Define a non-trivial irreflexive symmetric relation R over the set of natural numbers, showing why it has those properties. [3 marks]

(*c*) If A is a finite set with n elements, how many distinct irreflexive symmetric relations over A are there? [3 marks]

(*d*) If A is a finite set with n elements, how many distinct relations that are symmetric and antisymmetric over A are there? [3 marks]

(*e*) Suppose R and S are irreflexive symmetric relations over A. For each of the following relations, either prove that they are irreflexive and symmetric or give a counterexample.

(*i*) $R \cup S$

(*ii*) $R; S$

(*iii*) the relation Q defined to be

$$\{(X, Y) \mid X \subseteq A \wedge Y \subseteq A \wedge \forall x \in X. \forall y \in Y. (x, y) \in R\}$$

[8 marks]

SECTION C

5 Algorithms I

(*a*) State the *five* invariants of Red–Black Trees and briefly explain the advantages and disadvantages of Red–Black Trees over Binary Search Trees. [4 marks]

(*b*) For each of the possible types of 2–3–4-tree nodes, draw an isomorphic node cluster made of Red–Black nodes. The node clusters you produce must have the same number of keys and external links as the 2–3–4 nodes they replace and they must respect all the Red–Black tree rules when composed with other node clusters. [3 marks]

(*c*) What are the minimum and maximum possible number of nodes of a Red–Black tree with black-height h? Justify your answer. [4 marks]

(*d*) Explain, with clear pictures, what a "rotation" operation is, in the context of Binary Search Trees. [2 marks]

(*e*) Give a procedure for reshaping an arbitrary n-node Binary Search Tree containing n distinct keys into any other arbitrary Binary Search Tree with the same keys. [7 marks]

6 Algorithms I

(*a*) State the defining properties of a min-heap. Show how to convert between the tree and the (zero-based) array representation of a min-heap. [3 marks]

(*b*) "An array sorted in ascending order is always a min-heap." True or false? If false, offer a counter-example; otherwise, prove the correctness of this statement with respect to the defining properties of a min-heap you listed in response to part (*a*). [3 marks]

(*c*) The array

```
A I E R P M S N L
```

is not a min-heap. Why? Redraw it as a binary tree and turn it into a heap using the $O(n)$ `heapify()` procedure normally used as part of heapsort. Draw the intermediate stages as you go along and add any necessary explanations so that a reader can follow what you are doing and why. [4 marks]

(*d*) Perform `extractMin()` on the min-heap you produced in part (*c*). As before, draw the intermediate stages and add explanations as necessary. [3 marks]

(*e*) What is the asymptotic running time of the heapsort algorithm on an array of length n that is already sorted in ascending order? Justify your answer. [3 marks]

(*f*) What is the asymptotic running time of the heapsort algorithm on an array of length n that is already sorted in *descending* order? Justify your answer. [4 marks]

SECTION D

7 Software Design

(*a*) Describe the difference between a class and an instance. Show typical examples of each as they would be represented in UML diagrams and in source code. [4 marks]

(*b*) Explain the relationship between information hiding and loose coupling. Your explanation should mention class interfaces, visibility modifiers, and accessor methods. [6 marks]

(*c*) Consider the design of a future student records database system for Cambridge, and in particular, a module for examination registration and grading. Use this example to illustrate some of the separate phases of a software design project, by showing outline examples of:

(*i*) a usage scenario,

(*ii*) a class diagram related to that scenario,

(*iii*) a collaboration diagram related to the scenario, and

(*iv*) the public interfaces and fields for two of the classes in these diagrams.

You should not attempt to describe a complete design, but simply include enough detail to show the differences and relationships between these kinds of design model.

[10 marks]

8 Programming Methods and Java

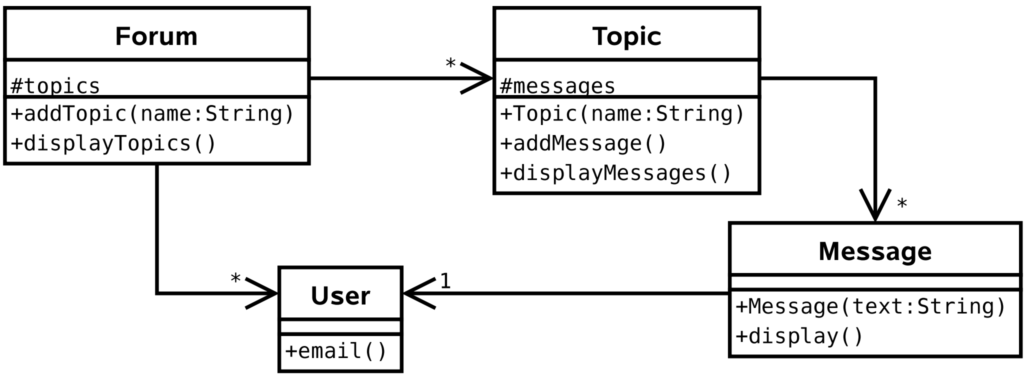
(*a*) Consider the method `sum` defined in Java as

```
public byte sum (int n) {
  byte result=0;
  for (int i=0;i<n;i++) result+=(1<<i);
  return result;
}
```

(*i*) What are the most positive and most negative values that can be represented by a `byte`? [2 marks]

(*ii*) Assuming `byte` and `int` are of arbitrary size, write down an expression for the result of `sum` in terms of `n`. For what range of `n` does `sum` actually compute the result of your expression? [4 marks]

(*b*) The UML class diagram below defines the software structure for an online forum, where users can post messages beneath specific topics. Topics are unique and displayed alphabetically, with associated messages displayed chronologically by creation date.



(*i*) Identify appropriate data structures from the Java standard library for the variables `topics` (the set of all topics) and `messages` (the set of all messages in a given topic). [4 marks]

(*ii*) Based on your choices, write Java code for the methods `addTopic` and `displayMessages`. [4 marks]

(*iii*) Using a design pattern, modify the UML diagram to allow users to "subscribe" to a topic (i.e. be notified using the method `email` when new messages are posted in that topic). Users should also be able to unsubscribe. You should identify the pattern and annotate any important methods. [6 marks]

9 Floating-Point Computation

(*a*) Briefly describe the 32-bit IEEE floating-point format, explaining what values (or other mathematical objects) are represented by bit-patterns in this format (you need not give the values corresponding to denormalised numbers). [4 marks]

(*b*) What value, if any, does the following Java method return, assuming `x` and `old` are held as 32-bit IEEE values?

```
float c() { float old=0, x=1;
            while (old != x) { old = x; x = x+1; }
            return x; }
```

Explain your reasoning. [3 marks]

(*c*) Consider the function computed by the Java method

```
float f(float x) { return x+1; }
```

Discuss how the use of 32-bit IEEE floating-point arithmetic causes it to differ from the mathematical function $f(x) = x + 1$. [4 marks]

(*d*) Given a problem of the form "find x such that $f(x) = y$", explain informally what it means for it to be *ill-conditioned*. [2 marks]

(*e*) The Newton–Raphson iteration for $\sqrt{a}$ uses $x_{n+1} = (x_n + a/x_n)/2$. Let $x_n = \sqrt{a} + \epsilon_n$, where the error ϵ_n is assumed to be small.

(*i*) Calculate how the error declines from one iteration to the next. [3 marks]

(*ii*) Given $1 \leq a < 4$ and $x_0 = 1.5$, how many iterations are necessary to achieve approximate 32-bit IEEE accuracy, and 64-bit IEEE accuracy? [2 marks]

(*iii*) Summarise a possible implementation of square-root on the whole 32-bit IEEE input range rather than just on $[1, 4)$. [2 marks]

END OF PAPER