A27217 *No Calculator permitted in this examination*

UNIVERSITY OF BIRMINGHAM

School of Computer Science

First Year – BSc in Artificial Intelligence and Computer Science
First Year – BSc in Artificial Intelligence and Computer Science with Industrial Year
First Year – BSc Natural Sciences
First Year – BA Computer Studies and French
First Year – BSc Computer Science
First Year – BSc Computer Science with Industrial Year
First Year – MEng Computer Science/Software Engineering
First Year – MEng Computer Science/Software Engineering with Industrial Year
First Year – BSc Computer Science with Business Management
First Year – BSc Computer Science with Business Management with Industrial Year
First Year – BSc Mathematics and Computer Science
First Year – BSc Mathematics and Computer Science with Industrial Year
First Year – BSc Pure Mathematics and Computer Science with Industrial Year

06 19339

Foundations of Computer Science

Summer Examinations 2013

Time allowed: 3:00 min

[Answer ALL Questions]

[Answer Each Part in a Separate Answer Book]

Part A

[Use a Separate Answer Book for THIS Part]
[Answer ALL Questions]

1. (a) Define the function `'append'` which appends two lists.
Example: `append [1,2,3] [1,2,3,4] = [1,2,3,1,2,3,4]`
[5%]

(b) Show, step by step, the evaluation of `append [1;2] [3]`. [3%]

(c) Prove that for any list `lst, append lst [] = lst`. [5%]

2. (a) Explain, in English, the differences between the mathematical type of integers and the OCaml integer type. [3%]

(b) How would you define the type of (mathematical, unrestricted) natural numbers in OCaml? Give your answer both in English and as an OCaml type. [5%]

(c) Implement subtraction (as a function called `minus`) for (mathematical, unrestricted) natural numbers. [5%]

(d) In OCaml, if `x, y, z` are variables of type `float`, is it true that `(x+y)+z = x +(y+z)`? Why? Give an example. [2%]

(e) The following function sums the elements of a list of integers:

```
let rec sum = function
| [] -> 0
| hd::tl -> hd + (sum tl)
```

Would you use it to sum a list of floating point numbers? Justify your answer. [2%]

3. (a) Define the function `select` which finds all the elements in a list satisfying a certain property (predicate), given as an argument.
Example:

```
let p x = x > 3
select p [1,2,3,4,5,6] = [4,5,6]
```

[5%]

(b) Prove that for any list `lst`,
`select (fun x -> true) lst = lst` [5%]

(c) Prove that for any predicate `p`,
`select p [] = []` [3%]

(d) Prove that for any list `lst` and any predicate `p`,
`select p (select p lst) = select p lst` [7%]

Part B

[Use a Separate Answer Book for THIS Part]
[Answer ALL Questions]

4. (a) Give a formal inductive/recursive definition of a *Binary Tree*, and then specify what additional conditions must be satisfied for a given *Binary Tree* to be (i) Perfectly *Balanced*, and (ii) a *Binary Search Tree*. [3%]

   (b) Draw the binary search tree that results from inserting the items [19, 11, 15, 4, 27, 22, 9, 12, 24, 29, 13] in that order into an initially empty tree. [3%]

   (c) State in words an efficient algorithm for deleting a given node from an existing binary search tree. [3%]

   (d) Comment on the time complexity of searching for items stored in a binary search tree compared with that for storage in an unsorted array. [3%]

5. (a) Describe in words how the *Mergesort* algorithm sorts an array of items. [3%]

   (b) What are the average-case and worst-case time complexities of *Mergesort* in terms of the number of items *n* to be sorted? Explain why. [3%]

   (c) Write an efficient pseudocode procedure `dif1(a)` that returns `true` if integer array `a` contains at least one pair of items that differ by no more than one, and `false` otherwise. For example, the array [1, 5, 1] returns `true` because 1 and 1 differ by 0, [2, 5, 1] returns `true` because 2 and 1 differ by 1, but [3, 5, 1] returns `false` because all pairs of items differ by at least 2. You may assume that you have access to a procedure `size(a)` that returns the size of an array `a`, and a procedure `mergesort(a)` that returns a sorted version of array `a` using the mergesort algorithm. You may also assume that you care more about the worst-case time complexity than the average-case time complexity. [3%]

   (d) Explain why it is relevant to know that you care more about the worst case rather than the average case. [3%]

6. (a) Explain what is meant by the terms *hash table*, *primary hash function*, *hash collision and direct chaining*. [4%]

   (b) How do the time costs of *lookups*, *insertions* and *deletions* for a good hash table depend on the number of hash table entries? Comment on the general efficiency of hash tables in terms of time and space complexity . [3%]

   (c) Suppose a small number of six digit student IDs are to be stored in a hash table represented as an array of size 11. The primary hash function is simply the first digit. Why is that not a sensible choice of hash function? Draw the initially empty hash table and insert the following keys into it using a secondary hash function that is simply the last digit plus 1: "063982", "120781", "149870", "093573", "125834"and "201938". [4%]

   (d) Suggest an improved primary hash function for the above type of data and explain why it would be better. [2%]

7. (a) Suppose you are given a graph specified by a symmetric $N \times N$ weight matrix. What does the symmetry of that matrix tell you about that graph? If M is the number of ∞ symbols in the matrix, i.e. absent connections, what is the connectivity proportion of the graph as a function of N and M? Draw the graph corresponding to the following weight matrix. [4%]

	A	*B*	*C*	*D*	*E*	*F*
A	0	4	3	∞	9	8
B	4	0	5	7	∞	∞
C	3	5	0	2	∞	7
D	∞	7	2	0	6	∞
E	9	∞	∞	6	0	1
F	8	∞	7	∞	1	0

   (b) Describe an efficient greedy vertex-based algorithm for determining a minimal spanning tree of a weighted graph. In what sense is your algorithm greedy? [3%]

   (c) What is the time complexity of your algorithm? Comment on how the graph's connectivity proportion affects the relative speed of your algorithm compared to Kruskal's edge-based algorithm for the same problem. [3%]

   (d) Use your algorithm to generate a minimal spanning tree starting from vertex A of the weighted graph specified above. Show the vertices and chosen edges after each step. [3%]