Q1: Algorithm design

This question asked for three algorithms for different tasks, with marks allocated for efficient solutions as well as complexity analysis.

It was answered well by most students, with an average well over 60%, and very few students getting less than 40%. Most students offered various techniques to provide efficient solutions - some quite inventive. Well done! The main areas where marks were lost were in offering incorrect or incomplete or inefficient algorithms, and in faulty or incomplete complexity analysis (esp not properly explaining the complexities).

Part (a): anagrams: Most students offered a solution based on sorting (we need to know which sorting algorithm is involved) or on hash-counting.

Part (b): symmetric difference of lists: Sorting of the lists helps, but one has to be careful in the way that the symm difference is extracted, especially the case when elements are equal, or one of the lists is empty. Hashing helps too, but some students added elements of only one list to the result.

Part (c): random selection of items. Some students provided clever solutions to this. However many provided possibly non-terminating algorithms (eg repeated selection from all items in the original list), and quite a few solutions were incomplete.

Question 2 on Sorting

The question was taken by 135 students.

The mean score was 14.3/20 with standard deviation 4.0.

Part (a) which asked for the worst case complexity of six different sorting algorithms was answered well by the great majority of students.

Part (b) which asked for definitions of common terms used to describe types of sorting behaviour or classes of algorithm was answered well, although some marks were dropped for a lack of detail in most cases.

Part (c) about a specialist compare function elicited more variable answers. Some showed good understanding of what a compare function is, and wrote correct pseudocode for it. Others were confused and wrote pseudocode for a whole sorting method. This was rarely correct in any case, and was uncalled for.

Part (c)(ii) asked for the output of a single merge operation of two already sorted lists, using the order defined in the compare function. About half the answers were precisely correct. The rest ranged from being completely incorrect, to being largely right with one or two small errors.

Although there is evidence that a small number of students have not picked up this topic adequately, the results on this question are pleasing overall.

Q3 was concerned with the algorithm complexity. The first part was a multiple choice type assessment, while the second part required that the student provide some rationale behind their choices. Vast majority of the students who took the question did very well, with almost 95% of the answers scoring 10+ marks. There were some very comprehensive and well thought answers. Overall, a very satisfactory result, and a clear sign that the students understand this important and difficult part of the course material.