9 Algorithms (RKH/DJW)

The *chaining* collision-resolution scheme for hash tables uses an array where each element is a linked list.

(*a*) For a hash table that uses chaining to resolve collisions with an array of length m and n keys inserted, the average asymptotic complexity for search and insert is $O(1 + \frac{n}{m})$.

(*i*) Explain why search and insert are considered to be $O(1)$ and not $O(n)$ in practice. [2 marks]

(*ii*) Give the worst-case complexities for search and insert. State the factors that determine the performance observed in practice. [5 marks]

(*b*) An alternative hash-table implementation replaces the linked lists in each slot with red-black trees. Discuss the advantages and disadvantages of this change. [4 marks]

(*c*) The linked lists may also be replaced by dynamically sized arrays. When full, an array is expanded by a factor of k.

(*i*) Derive a *potential* to compute the amortised cost of adding an element to such an array using the potential method. On inserting an element that triggers an expansion, your potential should be zero just after the expansion but before the insertion of the new element. [3 marks]

(*ii*) Use your potential to compute the amortised cost of adding an element to the array. [3 marks]

(*iii*) What factors would influence your choice of k when using the arrays in a hash table? [3 marks]