Homework 7

The exercises do *not* requite long writings. Try to be precise and to the point. Your answers should be short.

1. A dynamic array is a data structure that maintains an array and allows insertion and deletion operations of elements. As opposed to the conventional (static) arrays which have fixed and pre-determined lengths, for a dynamic array, an arbitrary number of elements can be inserted and thus the data structure needs to be able to grow in length. A common implementation of a dynamic array involves storing elements in a back-end array T. Whenever the number of elements in the array exceeds $\lambda \cdot T.\mathsf{length}$ where $\lambda \in (0, 1]$ is the load factor, the data structure creates a new empty array T' with twice the length of the current array, and places all elements in T contiguously at the start of T', before setting T' as the new T. In this way the back-end array T "grows" creating more capacity for new elements.
   Your task is to explain the following fact: Starting from the empty array, suppose we perform a sequence of m insertion operations. The total running time of these operations is $O(m)$. This means that the average running time of inserting an element into a dynamic array over a sequence of operations is $O(1)$.
2. Suppose the length w of integers in the universe is bounded by 8 and there are 16 buckets. Suppose we insert $142, 9, 204, 57, 43, 158, 201, 198, 89, 15, 177, 59$ using hash function $h(x) = x \bmod 16$, show the resulting hash table with
   (a) chaining
   (b) linear probing
   (c) double hashing with second hash function $h_2(x) = 7 - (x \bmod 7)$
3. The *text pattern matching* problem aims to find the first occurrence of a pattern string $p = p[0]p[1] \dots p[\ell - 1]$ in a long document $A = A[0]A[1]A[2] \dots A[n - 1]$. A simple way to solve this problem is to examine length-ℓ substrings of A of the form $A[i]A[i+1] \dots A[i+\ell-1]$ where $0 \leq i \leq n - \ell$ and compare them with the input pattern p. This procedure will take time $O(\ell n)$. We can improve the running time by utilising hashing. Suppose we use the hash function $h(s) = (s[0] + s[1] + \cdots + s[\ell - 1]) \bmod 2^d$. The procedure first computes the hash code $h(p)$. Then it compares $h(p)$ with the hash values of substrings $A[0] \dots A[\ell - 1], A[1] \dots A[\ell], A[2] \dots A[\ell + 1]$ and so on. If we have a match of hash values, then the algorithm compares the pattern string p with that substring character by character to verify the match. The algorithm returns the substring if it does matches with p, and it continues if the match is false. Show that this procedure can take time $O(\ell + n)$ plus the time spent refuting false matches.
4. Chris commute by train each morning from Manukau to Britomart. 90% of trains departing Manukau station on time. 80% of trains arriving in Britomart on time. 75% of trains depart on time and arrive on time.
   (a) Chris takes a train that departs on time. What is the probability that it will arrive on time?
   (b) Chris arrives in Britomart on time. What is the probability that his train departed on time?
   (c) Are the events, departing on time and arriving on time, independent?