1. Show that for any collection of hash function $H$, there exists $x, y$ such that

$$\sum_{h \in H} \delta_h(x, y) \geq |H| \left( \frac{1}{m} - \frac{1}{n} \right)$$

where $n$ and $m$ are the sizes of universe and table respectively.

2. Let $U$ be a universe of all possible keys of size $N$. Then prove that the size of a set of perfect hash functions that map $n$ keys into a table of $(m \geq n)$ is at least

$$\frac{C(N, n)}{(N/n)^n \cdot C(m, n)}$$

Here $C(a, b)$ denotes number of choices of $b$ subsets from a set of $a$ objects.

3. If $a, b$ are chosen uniformly at random then show that the hash function $h(x) = (ax + b) \mod p$ maps $x \neq 0$ uniformly at random to one of the $p$ values, i.e. probability $(h(x) = i) = 1/p$ for $0 \leq i \leq p-1, p$ is prime. Further show that for a pair of elements $x, y$, probability $(h(x) = i, h(y) = j) = probability(h(x) = i).probability(h(y) = j)$.

4. Let $|T| = p$ where $p$ is a prime. Define a hash function from $U = p^k$ to $T$ as follows. For a key $x = \langle s_1, s_2, \ldots, s_k \rangle$ with $0 \leq s_i \leq p-1$, and $a = \langle a_1, a_2, \ldots, a_k \rangle$, $a_i < p$, the hash function $h_a(x) = \sum_i a_i \cdot s_i \mod p$.

Prove that $h_a$ defines a strongly universal hash family.

5. Suppose $T$ is an ordered table of $n$ keys $x_i$, $1 \leq i \leq n$ drawn uniformly from $(0, 1)$. Instead of doing the conventional binary search, we use the following approach.

Given key $y$, we make the first probe at the position $s_1 = \lceil y \cdot n \rceil$. If $y = x_{s_1}$, we are through. Else if $y > x_{s_1}$, we recursively search for $y$ among the keys $(x_{s_1} \ldots x_n)$

else recursively search for $y$ among the keys $(x_1 \ldots x_{s_1})$.

At any stage when we search for $y$ in a range $(x_l \ldots x_r$, we probe the position $l + \lceil \frac{(y-x_l)(r-l)}{x_r-x_l} \rceil$.

We are interested in determining the expected number of probes required by the above searching algorithm.
In order to somewhat simplify the analysis, we modify the algorithm as follows. In round $i$, we partition the input into $n^{1/2^i}$ sized blocks and try to locate the block that contains $y$ and recursively search within that block. In the $i$-th round, if the block containing $y$ is $(x_l \ldots x_r)$, then we probe the position $s_i = l + \lceil \frac{(y-x_l)(r-l)}{x_r-x_l} \rceil$. We then try to locate the $n^{1/2^i}$-sized block by sequentially probing every $n^{1/2^i}$-th element starting from $s_i$.

Analyze the expected number of probes required. (Analyze the expected number of probes in each round using Chebychev’s inequality).

6. Given a set of points on the real-line with coordinates $x_1, x_2 \ldots x_n$, we want to determine if there is a subset $x_i, x_{i+1} \ldots x_{i+m-1}$ with separation distances $d_i \leq m$. Design a $O(n)$ algorithm for this problem.

7. Given a set $S$ of $n$ points in a plane - design a linear time algorithm to find the smallest enclosing circle of $S$.

8. Show how to implement the contraction algorithm (the original $n$ contractions) in

   (i) $O(n^2)$ time. You may want to use a adjacency matrix representation. To choose a random edge first choose a random vertex with probability proportional to its degree and then choose one of the neighbours at random. Prove that this works as required.

   (ii) $O(m \log n)$ time.

Extend the solution of the first part to weighted graphs.