Homework 4
Due: Wednesday, December 8 in LaTeX

1: Disk intersection

A *disk* consists of a circle plus its interior and is represented by its center point and radius. Two disks intersect if they have any point in common. Give an $O(n \log n)$ time algorithm to determine whether any two disks in a set of n intersect.

2: Edit distance

Given a string $s \in \Sigma^*$, where Σ is some finite alphabet, it is possible to perform the following string edit operations:

- $\mathsf{insert}(s, i, \alpha)$ returns the string $s[1..i-1]\alpha s[i..|s|]$, where $1 \le i \le |s|+1$ and $\alpha \in \Sigma$.
- $\mathsf{delete}(s, i)$ returns the string $s[1..i-1]s[i+1..|s|]$,
- $\mathsf{substitute}(s, i, \alpha)$ returns the string $s[1..i-1]\alpha s[i+1..|s|]$.

The *edit distance* $\delta(s, t)$ between two strings s and t is the minimal number of edit operations required to transform s to t. The edit distance $\delta(s, t)$ can be computed by the following recursive formula:

$$\delta(s,t) = \begin{cases} |s| & \text{if } \ |t| = 0 \\ |t| & \text{if } \ |s| = 0 \\ \min\{1+\delta(s,t[2..|t|]), 1+\delta(s[2..|s|],t), 1+\delta(s[2..|s|],t[2..|t|])\} & \text{if } \ s[1] \neq t[1] \\ \min\{1+\delta(s,t[2..|t|]), 1+\delta(s[2..|s|],t), \delta(s[2..|s|],t[2..|t|])\} & \text{if } \ s[1] = t[1] \end{cases}$$

a) Show how to compute $\delta(s, t)$ in time $O(|s| \cdot |t|)$ and additional space $O(\min\{|s|, |t|\})$.

b) Show how to compute a sequence of edit operations of length $\delta(s, t)$ which transforms s to t in time $O(|s| \cdot |t|)$ and additional space $O(\max\{|s|, |t|\})$.

3: Minimal perfect hash funtions *

Let $U = \{0, 1\}^w$ be a universe of binary strings for some $w \in I\!N$, and let $S \subseteq U$ and $n = |S|$. A *perfect hash function* for S with range r is a mapping $h : U \to \{1, 2, \ldots, r\}$ such that h is one-to-one on S, where $r \ge n$. A *minimal perfect hash function* for S is a perfect hash function for S with range n. The FKS perfect hash function uses $O(n)$ words of memory each of size $O(\log |U|)$ bits to store bucket sizes and secondary hash functions, i.e. the FKS perfect hash function uses space $O(n \log |U|)$ bits. The goal of this exercise is to improve the space bound to $O(n + \log |U|)$ bits, while maintaining constant evaluation time. To simplify things we will work in the *cell probe model*, that is, any function on an $O(\log |U|)$ bit memory word can be computed in constant time.

a) Show that using space $O(\log |U|)$ bits it can be assumed that the universe has size $O(n^2)$. (Hint: Use universal hash functions).

b) Show that the secondary hash functions can be derived from a small set of universal hash functions $\{h_1, \ldots, h_{\log n}\}$ (depending on the set S) such that: Half of the secondary hash functions are h_1 (modulo the range), one quarter of the secondary hash functions are h_2 (modulo the range), etc. (Hint: Increase the range of the secondary hash functions).

c) Design an auxiliary data structure which can store a set of nonnegative integers $z_1, \ldots, z_k$, where $z_i \le n$, that uses space $O(k + \sum_{i=1}^{k} z_i)$ bits, and supports constant time prefix sum queries, i.e. given some j report the sum $\sum_{i=1}^{j} z_i$.

d) By combining b) with several applications of c) show how to construct minimal perfect hash functions using space $O(n + \log |U|)$ bits.