CS6033 Homework Assignment 3*

Due Feb. 14th at 5:30 p.m.
Turn in this assignment as a PDF file on NYU classes
No late assignments accepted†

1. (15 points) "Imagine that you work for an insurance company that is insuring people against identity theft. You have just learned about a major security breach at a prominent bank used by many of your customers. Through back channels, you have obtained the list of Social Security numbers of the bank customers whose banking records were stolen, and, of course, you know the Social Security numbers for your own customers. Describe an efficient scheme for identifying which of your customers were victims in this security breach."

   What is the running time of your method in terms of the number of customers of your insurance company and the number of bank customers who were victims in this security breach? Use Theta notation. Your algorithm must be efficient.

2. (15 points) "One way to measure the reading difficulty of a book is to count the number of unique words it contains. For example, Green Eggs and Ham, by Dr. Seuss, contains 50 unique words, whereas the book of Isaiah, from the Bible, contains almost $2,000$ unique (Hebrew) words. Suppose you have a book, B, containing n total words (including duplicates). Describe an efficient scheme to count the number of unique words in B. You may assume that you have a parser that returns the n words in B as a sequence of character strings in $O(n)$ time."

   Your algorithm must be efficient. What is the running time of your method in Theta notation?

3. (15 points) Your boss asked you to store the customer id's so they can be quickly inserted, deleted for found. You decide to use a hash table. The expected number of customer id's is 100. You therefore choose a hash table of size 100. The customer ID's range uniformly from 0 to 20200. Your co-workers have suggested the following hash functions:

   - $h(i) = i^2 \mod 100$
   - $h(i) = i^3 \mod 100$
   - $h(i) = (11 * i^2) \mod 100$
   - $h(i) = (12 * i) \mod 100$

   It falls on you to save your team from the brink of destruction by deciding which coworker will implement their idea. (You feel to lazy to do anything yourself.) What shall you do?

4. (10 points) How many times must you attempt to construct one of a perfect hash table's sub-tables so you have one with probability greater than 99.9999%? This is a failure rate of 1 out of one million.

*Many of these questions came from outside sources.
†If the class has started, your homework will be late and will not be accepted for credit.

5. Consider the following alternative implementation of a hash table.

   The hash table A is implemented as an array of size m storing binary values (i.e. each slot contains only 0 or 1). Initially every slot is set to 0. You are able to access every single slot in $O(1)$ time. There are two operations `INSERT` and `CHECK` that can be performed on the hash table A:

   - `INSERT`(k): this operation inserts k into A, i.e. sets $A[h(k)]$ to 1 in $O(1)$ time where $h(k)$ is the hash function.
   - `CHECK`(k): this operation checks if k is already in hash table A. i.e. if $A[h(k)] == 1$, this operation returns true. Otherwise, it returns false. This operation takes $O(1)$ time.

   (a) (15 points) What is the probability `CHECK`(k) returns the wrong answer? i.e. k does not exist in A, but `CHECK`(k) returns true. (Hint: the probability of a collision occurring with two keys.)

   Assume the number of items already in the hash table A is n.

   (b) (15 points) If you're given another hash function $h'(x)$, how can you reduce the probability that `CHECK`(k) returns the wrong answer? Explain how you can re-design the `INSERT`(k) and `CHECK`(k) operations, WITHOUT using any extra space.

   For your re-designed hash table what is the probability that `CHECK`(k) returns the wrong answer?

6. (15 points) Consider the following family of hash functions:

$$H = \{h_{a,b}(x) = (ax + b \texttt{ mod } n) \texttt{ mod } m \texttt{ where } a \in \{1, ..., n-1\} \texttt{ and } b \in \{0, ..., n-1\}\}$$

   where $m = 2n$.

   Is H universal? If so, prove H is universal. If not, provide a counter example to prove it is not universal.

7. (3 bonus points) Think of a <u>*good*</u>[1] exam question for the material covered in Lecture 2

[1] Only well thought out questions will receive the bonus points.