1 Reading.

Chapter 5 and the supplemental reading on hash tables. (If you didn’t get one in class, you can find it on Blackboard.)

2 Problems.

1. Problem 5.1.

2. In the balls-and-bins example in class, two balls collide when if they are tossed into the same bin. This problem concerns counting the total number of collisions. If three distinct balls $a$, $b$, and $c$, collide at the same location it counts as $\binom{3}{2} = 3$ collisions (as $a$ collides with $b$, $b$ collides with $c$, and $a$ collides with $c$; if four balls collide at the same location it counts as $\binom{4}{2}$ collisions, etc. Consider the following question about tossing $n$ balls in to $s$ bins. Express your answers terms of $s$ and $n$, as needed.

(a) For an arbitrary (i.e., not random) tossing of the balls into bins, what is the worst case number of collisions on $n$ keys? Give an example that achieves this worst case.

(b) Suppose we toss the balls uniformly at random. What is the expected total number of collisions? Prove your bound. (Hint: define the appropriate set of indicator variables and use linearity of expectation.)

3. Suppose we were to implement hash tables via the bit array representation discussed in class, i.e., where $\text{bitarray}[k] = 1$ when key $k$ is in the table and $\text{bitarray}[k] = 0$ when the $k$ is not in the table. Such a hashtable can be used to keep track of a dictionary that contains any subset of the $N$ possible keys $\{0, \ldots, N-1\}$. Describe how to implement such a table with the operations: create, insert, delete, and find. All operations should be $\Theta(1)$. In particular create should be $\Theta(1)$. You may assume that allocating an uninitialized array can be done in $\Theta(1)$; however, this array is uninitialized. Clearly, you cannot zero all the elements in such an uninitialized array in $\Theta(1)$. Hint: you may maintain auxiliary data structures in addition to a length $N$ array of bits.