Notes: Please work on this with your group-mate(s); just submit one writeup per group. Consulting other sources (including the Web) is not allowed. Write your solutions neatly; if you are able to make partial progress by making some additional assumptions, then state these assumptions clearly and submit your partial solution.

1. Recall the construction for \( n \) pairwise independent bits that we saw in class: assume \( n \) is a power of two, and for \( 0 \leq i \leq n - 1 \), let \( b_i \) be the \((\log n + 1)\)-bit string obtained by concatenating the bit “1” to the end of the \( \log n \)-bit binary representation of \( i \); now, for a random \((\log n + 1)\)-bit random string \( r \), define each \( X_i \) to be \( b_i \cdot r \mod 2 \).

Prove that these random bits \( X_0, X_1, \ldots, X_{n-1} \) are actually three-wise independent. (7 points)

2. One of the basic results we showed using the probabilistic method was that for all \( n \) large enough (say, \( n \geq 10 \)), there exist \( n \)-vertex graphs \( G \) with no clique or independent set of size more than \( 2 \log_2 n \). Given \( n \), develop a deterministic algorithm running in time \( 2^{O(\log^2 n)} \) to construct such a graph. (5 points)

3. We have a hash table \( T \) implemented as an array of \( m \) linked lists: each of \( T[0], T[1], \ldots, T[m-1] \) is a pointer to the head of a linked list. To insert an element \( x \) that hashes to \( i \) (for some \( i \in \{0, 1, \ldots, m-1\} \), under some given hash function), we will do the following: do a standard search in the linked list pointed to by \( T[i] \); insert \( x \) at the end of this list iff \( x \) was not found in the list. The traversal of each element of the linked list takes unit time.

Let \( h \) be a random hash function mapping the set \( A = \{0, 1, \ldots, n - 1\} \) to \( B = \{0, 1, \ldots, m - 1\} \), such that each \( h(i) \) is uniformly distributed in \( B \), and that the random variables \( \{h(x) : x \in A\} \) are pairwise independent. Suppose \( a \) distinct elements of \( A \) have been inserted into \( T \) thus far, and that we now want to insert an element \( x \) into \( T \). \( x \) may or may not be one of the \( a \) elements already inserted. What is the worst-case expected running time for inserting \( x \), given the model for traversing \( T \) and measuring running-time from the previous paragraph? (The worst case is over the worst possible choice of the \( a \) elements and \( x \); the expectation is over the random choice of \( h \).) (5 points)

4. Read the first algorithm of the Bar-Yossef et al paper that we studied in class, from www.ee.technion.ac.il/people/zivby/papers/f0/f0.ps. We will assume that you have full familiarity with this algorithm.

Does the paper’s analysis change by much if we use Chebyshev-Cantelli instead of Chebyshev? (3 points)