CS2241: Homework 1. Due March 26.

Please work on problems on this assignment completely on your own. (You may of course ask the professor or the TF for assistance, but avoid working with other students.) You may use standard mathematical software (Maple, Matlab, etc.) and you may write programs to help you determine answers. You should do 5 of the 7 problems; if you do 6, we'll use your top 5 scores.

1. Determine Pagerank scores and Hub and Authority scores for the following graph (given by the transition matrix). State whatever assumptions you make in calculating the scores (for example, what scaling methods you are using, if any).

	a	b	c	d	e	f
a	0	0	1	1	0	1
b	0	0	1	0	0	1
c	0	0	0	1	0	1
d	0	1	1	0	0	0
e	1	0	0	0	0	0
f	0	0	0	0	1	0

2. Suppose that I suggest the following alternative method for obtaining Hub and Authority scores, based on random walks. To obtain Authority scores, we consider the following random walk: from a page p_1, a step consists of first following back a random inlink from the current page to a page p_2, and then following forward a random outlink from this page to a page p_3. The step of the Markov chain takes us from p_1 to p_3. Similarly, Hub scores are computed as follows: from a page p_1, a step consists of first following forward a random outlink from the current page to a page p_2, and then following backward a random inlink from this page to a page p_3. The step of the Markov chain takes us from p_1 to p_3. We assume that these Markov chains are finite, irreducible, and aperiodic and hence have a unique stationary distribution.

Prove that, using this approach, the Hub score for a page is simply proportional to the number of outlinks and the Authority score is proportional to the number of inlinks. (Note: you will need the idea of an equilibrium distribution [of a Markov chain/random walk] for this problem; if you're uncomfortable with that, you may wish to do another problem.)

3. The following approach is often called *reservoir sampling*. (It is useful, naturally, in streaming algorithms; you should think about why this might be.) Suppose that we have a sequence of items, passing by one at a time. We want to maintain a sample of one item that has the property that it is uniformly distributed over all the items that we have seen at each step. Moreover, we want to accomplish this without knowing the total number of items in advance or storing all of the items that we see.

Consider the following algorithm, which stores just one item in memory at all times. When the first item appears, it is stored in the memory. When the k-th item appears, it replaces the item in memory with probability $1/k$. Explain why this algorithm solves the problem.

Now suppose instead we want a sample of s items instead of just one, *without replacement*. That is, we don't want to get the same item multiple times in our sample. (If this wasn't an issue, we could get a sample of s items with replacement just by runnig s independent copies of the above.) Generalize the above process to that case. (Hint: start by taking the first s items and storing them as your sample. With what probability should each new item come into the sample?)

4. A fair coin is flipped until the first head occurs. Let X denote the number of flips required. Find the entropy $H(X)$ in bits. Suppose your friend flips a fair coin until the first head is flipped to generate a value for X, and now you want to ask a series of yes-no questions (of the form "is X contained in the following set?") to determine the value of X generated. Describe what questions you ask, determine the expected number of questions you ask, and compare your result with $H(X)$.

5. Consider arithmetic coding for the string *abcbaab* when the probability of an a is 0.2, a b is 0.3, and a c is 0.5. Show each step for idealized arithmetic coding, with real number arithmetic. Now suppose someone gave you the real number 0.63215699. Decode a sequence of length 10 corresponding to the above model. For both problems, you should take the real interval $[0, 1]$ as broken up into subintervals $[0, 0.2]$,$[0.2, 0.5]$ and $[0.5, 1]$, and similarly recursively.

6. Compress the string "abracadabraarbadacarba" using the LZ77 approach (sliding windows), with a window size of six and a lookahead buffer of size six. Similarly, compress it using the LZ78 and LZW approaches. For the LZW approach, assume the letters a, b, c, d, and r start in the dictionary in alphabetical order.

7. For the following, let $d = 1024$. (You can use higher values of d if you wish to test your code, but provide results for $d = 1024$.) Write code that first generates a random point on the d-dimensional unit sphere, then rotates it using a Randomized Hadamard Transform, then quantizes coordinates to either -1 or 1 (not stochastic quantization, just round by sign), the does the inverse rotation, and then finds the mean squared error between the resulting point/vector and the original. Have the code run 100 times and give the min, mean, and max error. For this problem, you may use any libraries or tools (such as LLMs) to help generate the code as you wish.

Note: you do not have to turn in the code, just keep it handy. Provide the results for $d = 1024$, and a paragraph description of your code (any tools you used, any issues you had, etc.).