CS 473, Spring 2023
Homework 4 (due Mar 1 Wed 9pm)

Instructions: As in Homework 1.

Problem 4.1: In a popular form of logic puzzles, you land on an island that has three types of inhabitants: "knights", who always tell the truth; "knaves", who always lie; and "spies", who sometimes lie and sometimes tell the truth.

Suppose there are n inhabitants, where 60% are known to be decent folks, i.e., knights. The remaining 40% are bad, i.e., knaves or spies. You want to know who are the good/bad guys, i.e., determine the types of all n inhabitants. You are allowed to ask only questions of the form, "is person A a knight/knave/spy?", to another person B. (All the inhabitants know each other.) Obviously, if you can find a person who you know is a knight, the problem can be solved after asking n additional questions.

(a) (10 pts) Give a (very) efficient Monte Carlo algorithm that finds a knight. State the probability of error. (Hint: this is supposed to be very easy!)

(b) (90 pts) Give a Las Vegas algorithm that finds a knight by asking $O(n)$ expected number of questions. Analyze the constant factor in the big-Oh and make it smaller than 1.5.

(Hint: use (a). How can you confirm whether a specific person is a knight by asking $O(n)$ questions?)

(Note: there is also a deterministic algorithm that requires $O(n)$ questions, but it is more complicated and has a larger constant.)

Problem 4.2: We are given a set of n elements, where d of them are "defective" ($d < n/2$). We know the value of d in advance. We want to identify the defective elements. The only allowed operation is the following test: given a subset S, if S contains no defective elements, the tester reports "ok"; if S contains exactly one defective element, the tester reports this element; however, if S contains two or more defective elements, the tester reports "inconclusive".

(a) (20 pts) Give a deterministic algorithm that finds one defective element with $O(\log n)$ tests.

(Note: this is the best possible, as there is a matching lower bound for deterministic algorithms—you don't need to prove this.)

(b) (70 pts) Give a Las Vegas algorithm that finds one defective element with $O(1)$ expected number of tests.

(Hint: pick a subset where for each element, we decide to put it in the subset independently with probability $1/d$. It might be helpful to know that $\lim_{k\to\infty}(1 - 1/k)^k$ is a constant. . .)

(c) (10 pts) Give a Las Vegas algorithm that finds all defective elements with $O(d)$ expected number of tests (using (b)).

Problem 4.3: Given a string $s \in \Sigma^*$ of length n, and given an integer k, we want to find the longest substring t such that t occurs at least twice, and two occurrences are separated by at least k characters. (In other words, s contains tzt for some z of length at least k.)

For example, for $s = 1100100100100110$ and $k = 5$, one answer is 1001 (by writing $s =$ 1**1001**00100**1001**10).

Describe an efficient randomized Las Vegas algorithm for this problem, by using the Karp–Rabin fingerprint technique.

(Hint: first solve the decision problem: given ℓ, decide whether there exists a substring t of length ℓ satisfying the stated conditions... An algorithm with $O(n \log^c n)$ expected running time for some constant c will get full credit.)