Write your answers in the separate answer booklet.
Please return this question sheet and your cheat sheet with your answers.

---

1. For any positive integer n, the nth ***Fibonacci string*** F_n is defined recursively as follows, where $x \bullet y$ denotes the concatenation of strings x and y:

$$\begin{aligned} F_1 &:= \texttt{0} \\ F_2 &:= \texttt{1} \\ F_n &:= F_{n-1} \bullet F_{n-2} \quad \text{for all } n \geq 3 \end{aligned}$$

For example, $F_3 = \texttt{10}$ and $F_4 = \texttt{101}$.

(a) What is F_8?

(b) ***Prove*** that every Fibonacci string except F_1 starts with $\texttt{1}$.

(c) ***Prove*** that no Fibonacci string contains the substring $\texttt{00}$.

2. You have reached the inevitable point in the semester where it is no longer possible to finish all of your assigned work without pulling at least a few all-nighters. The problem is that pulling successive all-nighters will burn you out, so you need to pace yourself (or something).

Let's model the situation as follows. There are n days left in the semester. For simplicity, let's say you are taking one class, there are no weekends, there is an assignment due every single day until the end of the semester, and you will only work on an assignment the day before it is due. For each day i, you know two positive integers:

- $Score[i]$ is the score you will earn on the ith assignment if you do *not* pull an all-nighter the night before.
- $Bonus[i]$ is the number of additional points you could potentially earn if you *do* pull an all-nighter the night before.

However, pulling multiple all-nighters in a row has a price. If you turn in the ith assignment immediately after pulling k consecutive all-nighters, your actual score for that assignment will be $(Score[i] + Bonus[i])/2^{k-1}$.

Design and analyze an algorithm that computes the maximum total score you can achieve, given the arrays $Score[1..n]$ and $Bonus[1..n]$ as input.

3. The following algorithm finds the smallest element in an unsorted array. The subroutine Shuffle randomly permutes the input array A; every permutation of A is equally likely.

```
RandomMin(A[1..n]):
  min ← ∞
  Shuffle(A)
  for i ← 1 to n
      if A[i] < min
          min ← A[i]        (⋆)
  return min
```

In the following questions, assume all elements in the input array $A[\,]$ are distinct.

(a) In the worst case, how many times does RandomMin execute line $(\star)$?

(b) For each index i, let $X_i = 1$ if line $(\star)$ is executed in the ith iteration of the for loop, and let $X_i = 0$ otherwise. What is $\Pr[X_i = 1]$? *[Hint: First consider $i = 1$ and $i = n$.]*

(c) What is the *exact* expected number of executions of line $(\star)$?

(d) ***Prove*** that line $(\star)$ is executed $O(\log n)$ times with high probability, *assuming* the variables X_i are mutually independent.

(e) **[Extra credit]** ***Prove*** that the variables X_i are mutually independent.
[Hint: Finish the rest of the exam first!]

4. Your eight-year-old cousin Elmo decides to teach his favorite new card game to his baby sister Daisy. At the beginning of the game, n cards are dealt face up in a long row. Each card is worth some number of points, which may be positive, negative, or zero. Then Elmo and Daisy take turns removing either the leftmost or rightmost card from the row, until all the cards are gone. At each turn, each player can decide which of the two cards to take. When the game ends, the player that has collected the most points wins.

Daisy isn't old enough to get this whole "strategy" thing; she's just happy to play with her big brother. When it's her turn, she takes the either leftmost card or the rightmost card, each with probability $1/2$.

Elmo, on the other hand, *really* wants to win. Having never taken an algorithms class, he follows the obvious greedy strategy—when it's his turn, Elmo *always* takes the card with the higher point value.

Describe and analyze an algorithm to determine Elmo's expected score, given the initial sequence of n cards as input. Assume Elmo moves first, and that no two cards have the same value.

For example, suppose the initial cards have values $1, 4, 8, 2$. Elmo takes the 2, because it's larger than 1. Then Daisy takes either 1 or 8 with equal probability. If Daisy takes the 1, then Elmo takes the 8; if Daisy takes the 8, then Elmo takes the 4. Thus, Elmo's expected score is $2 + (8 + 4)/2 = 8$.