Задача 1-1 (5 баллов). Пусть $t_r(I)$ обозначает время алгоритмы рандомизированного алгоритма на входе I при значениях случайных бит r. Для заданного множества входов $\mathcal{I}$ рассмотрим две “меры сложности“:

$$A = \max_{I \in \mathcal{I}} E_r \left[t_r(I) \right], \qquad B = E_r \left[\max_{I \in \mathcal{I}} t_r(I) \right].$$

Можно ли утверждать, что одно из чисел A и B всегда не меньше другого?

Задача 1-2 (15 баллов). Рассмотрим произвольное (корректное) решающее дерево для задачи сортировки n ключей, в котором все листья являются достижимыми. Найдите точную (не асимптотическую!) нижнюю оценку на *наименьшую* из глубин листьев данного дерева, т.е. такую функцию $g(n)$, что в любом подобном дереве наименьшая глубина листа не меньше $g(n)$, и одновременно для любого n существует такое дерево, в котором наименьшая глубина листа равна $g(n)$.

Задача 1-3 (15 баллов). Рассмотрим задачу слияния двух упорядоченных последовательностей длины m и n, $m \geq 2n$. Докажите (используя модель решающего дерева) нижнюю оценку $\Omega(n \log \frac{m}{n})$ на количество сравнений, необходимых для решения данной задачи в худшем случае.

Задача 1-4 (30 баллов). Рассмотрим алгоритмическую задачу с конечным множеством входов $\mathcal{I}$ и произвольное вероятностное распределение на нем. Рассмотрим также некоторое конечное семейство $\mathcal{A}$ детерминированных алгоритмов для ее решения и вероятностное распределение на нем. Обозначим через $f_A(I)$ время работы алгоритма A на входе I.

1. Докажите неравенство

   $$\min_{A \in \mathcal{A}} E_I \left[f_A(I) \right] \leq \max_{I \in \mathcal{I}} E_A \left[f_A(I) \right],$$

   где матожидания взяты относительно соответствующих распределений. Сформулируйте данное неравенство в терминах соотношения сложности в среднем (по входам) и рандомизированной сложности.
2. Придумайте и сформулируйте определение *рандомизированного* алгоритма сортировки в модели решающих деревьев. Докажите в этой модели оценку $\Omega(n \log n)$ для сложности произвольного рандомизированного алгоритма сортировки, основанного на сравнении ключей.

Задача 1-5 (20 баллов). Предложите реализацию *стека* на основе (одного) массива, которая поддерживает операции добавления в конец и удаления из конца. Требуется, чтобы *емкость* (количество выделенных ячеек памяти) стека в любой момент времени отличалась от фактического размера не более чем в константу раз, а учетная сложность операций добавления в конец и удаления из конца была константной.

Задача 1-6 (20 баллов). Предложите реализацию *очереди* на основе (одного) массива, которая поддерживает операции добавления в конец и удаления из начала. Требуется, чтобы емкость очереди в любой момент времени отличалась от фактического размера не более чем в константу раз, а учетная сложность операций добавления в конец и удаления из начала была константной.

Задача 1-7 (30 баллов). Покажите, как *деамортизировать* операции вставки в конец вектора, т.е. добиться того, чтобы операции добавления в конец и чтения элемента по индексу требовали $O(1)$ времени в *худшем* случае. Совет: по мере добавления новых элементов необходимо параллельно копировать уже имеющийся массив в массив увеличенного размера. Делать это следует с такой скоростью, чтобы в тот момент, когда меньший массив окажется заполнен, мы могли за время $O(1)$ выполнить переключение на новый массив.