Задача 1-1 (15 баллов).

1. Докажите, что в бинарном дереве с k листьями высота составляет $\Omega(\log k)$.
2. Рассмотрим произвольное (корректное) решающее дерево для задачи сортировки n ключей, в котором все листья являются достижимыми. Найдите точную 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 1-2 (30 баллов). Рассмотрим алгоритмическую задачу с множеством входов $\mathcal{I}$ и произвольное вероятностное распределение на нем. Рассмотрим также некоторое семейство $\mathcal{A}$ детерминированных алгоритмов для ее решения и вероятностное распределение на нем. Обозначим также через $f_A(I)$ время работы алгоритма A на входе I.

1. Докажите неравенство

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

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

Задача 1-3 (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 1-4 (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 1-5 (25 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. Будем считать, что выделение и освобождение участка памяти произвольного размера требует $O(1)$ времени. 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.

Задача 1-6 (35 баллов). Выпишите рекуррентное соотношение для случайной величины $h(n)$, равной глубине рекурсии в алгоритме Quick-Sort для массива из n ключей при рандомизированном способе выбора разделяющего элемента. Докажите, что $E[h(n)] = O(\log n)$.