1. На вход задачи подаётся число n и последовательность целых чисел $a_1, \ldots, a_n$. Необходимо найти номера i и j $(1 \leqslant i < j \leqslant n)$, такие что сумма $\sum_{k=i}^{j} a_k$ максимальна.

1. Постройте линейный алгоритм, решающий задачу.

2. Постройте онлайн-алгоритм (достаточно выполнить только этот пункт).

2. Фирма производит программное обеспечение для банкоматов разных стран мира. Иногда, купюры какого-то вида заканчиваются и банкомату нужно определить, возможно ли выдать клиенту требуемую сумму. Вход задачи: число n, номиналы купюр $v_1, \ldots v_n$ и сумма клиента s.

1. Постройте алгоритм, решающий задачу за $O(ns)$.

2*. Постройте алгоритм, решающий задачу за $O(na \log a)$, где $a = \min a_i$.

3. Рассмотренный нами алгоритм вычисления расстояния редактирования строк длины m и n заполняет таблицу размера $O(mn)$. При больших m и n такому алгоритму просто не хватит памяти. Как можно обойтись меньшим объёмом памяти?

1. Допустим, что нас интересует только расстояние редактирования, но не соответсвующее оптимальное выравнивание. Покажите, что тогда в каждый момент не нужно хранить всю таблицу и можно обойтись объёмом памяти $O(n)$.

2. Теперь допустим, что мы хотим найти и оптимальное выравнивание. Легко видеть, что это эквивалентно поиску кратчайшего пути из вершины $(0, 0)$ в вершину (n, m) в соответсвующем графе: вершины графа — клетки таблицы, а стоимость ребра — 1 или 0, в зависимости от конкретного перехода. Любой такой путь должен проходить через вершину $(k, m/2)$ для некоторого k. Модифицируйте алгоритм поиска расстояния редактирования, чтобы он заодно выдавал и k. Считайте, что m — степень двойки.

3*. Рассмотрим теперь следующий рекурсивный алгоритм.

```
1 Function FindPath((0,0) → (n,m)) :
2 |   Вычислить k (см. предыдущий пункт);
3 |   FindPath((0,0) → (k,m/2));
4 |   FindPath((k,m/2) → (n,m));
5 |   return объединение найденных двух путей
6 end
```

Покажите, что по данной схеме алгоритм можно реализовать так, чтобы время его работы было $O(nm)$, а память — $O(n)$.

4 [Шень 2.1.1.]. Постройте алгоритм, который, получив на вход числа n и k, выводит все последовательности длины k целых чисел от 1 до n.

5. В одном языке программирования операция разрезания строки на две части реализовано через копирование. Разрезать строку $w = uv$ на две части u и v будет стоить длину строки $O(|w|)$, где $|w|$ — длина строки w, вне зависимости от длин строк u и v. Постройте алгоритм, который получив на вход строку $w = u_1 u_2 \ldots u_k$ и номера позиций

$i_1 = |u_1|, i_2 = |u|_1 + |u_2|, \ldots, i_k = |u_1| + \ldots + |u_k|$ строит последовательность разрезов строки w, которая приводит к разрезанию w на подстроки $u_1, \ldots, u_k$ и использует для этого минимальное число операций.

Заметьте, что в зависимости от порядка разрезов меняется общее число операций: если строку длины 20 нужно разрезать на строки с номерами позиций 3, 10, то отрезав сначала строку в позиции 3 будет затрачено $20 + 17 = 37$ операций, а отрезав сначала строку в позиции 10, будет потрачено $20 + 10 = 30$ операций.

6. Алиса и Боб играют в следующую игру. Есть n карт, на i-й карте записано число $1 \leqslant a_i \leqslant n$ (числа могут повторяться!).

Игроки ходят по очереди, первой ходит Алиса. На каждом ходу игрок выбирает карту и выбрасывает её. Кроме того, он выбрасывает все карты, на которых число меньше, чем на выбранной. Формально, если игрок выбирает карту с номером i, он выбрасывает эту карту, а также каждую карту с номером j, такую что $a_j < a_i$.

Игрок проигрывает, если он не может сделать ход, то есть если не осталось карт.

1. Постройте алгоритм, который определяет победителя, если игроки играют оптимально.

2. Постройте жадный алгоритм, решающий эту задачу.

7 [Шень 8.1.4]. Дан выпуклый n-угольник (заданный координатами своих вершин в порядке обхода). Его разрезают на треугольники диагоналями, для чего необходимо $n-2$ диагонали (это можно доказать индукцией по n). Стоимостью разрезания назовём сумму длин всех использованных диагоналей. Постройте полиномиальный алгоритм, который находит минимальную стоимость разрезания. (Перебор не подходит, так как число вариантов не ограничено многочленом.)