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 [Шень 1.3.3]. Даны две последовательности `x[1]` ... `x[n]` и `y[1]` ... `y[m]` целых чисел. Постройте алгоритм, который находит максимальную длину последовательности, являющейся подпоследовательностью обеих последовательностей. Сложность алгоритма $O(nm)$.

3. 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 $a = \min a_i$.

4. Вершинным покрытием (vertex cover) графа $G(V, E)$ называется множество его вершин $S \subseteq V$, которое содержит хотя бы один конец каждого ребра графа. Постройте линейный алгоритм, который находит минимальное (по числу вершин) вершинное покрытие для дерева (вход задачи).

5. 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 — $O(n)$.

6. Необходимо разбить набор целых чисел $a_1, \ldots, a_n$ на три части так, чтобы суммы чисел в каждой из частей были одинаковы. Формально, нужно найти разбиение $\{1, \ldots, n\} = I \cup J \cup K$ (множества I, J и K попарно не пересекаются), такое что

$$\sum_{i \in I} a_i = \sum_{j \in J} a_j = \sum_{k \in K} a_k = \frac{1}{3} \sum_{i=1}^{n} a_i.$$

Постройте алгоритм, основанный на динамическом программировании, со временем работы, зависящим полиномиально от n и $\sum\limits_{i=1}^{n} a_i$.