Задача A. Красота превыше всего

Давайте искать отрезок на котором есть все виды деревьев. Будем поддерживать массив количества вхождений каждого вида в поддерживаемый отрезок и количество нулей в этом массиве. Если отрезок не содержит всех типоп элементов, то двигаем правый указатель, иначе левый.

```
int l = 0, r = 0, ans = n;
vector<int> cnt(k);
int cnt_zero = k;
while (r != n || l != n) {
    if (cnt_zero == 0) {
        ans = min(ans, r - l);
        cnt[a[l]]--;
        if (cnt[a[l]] == 0) {
            cnt_zero++;
        }
        l++;
    } else {
        if (r == n) {
            break;
        }
        cnt[a[r]]++;
        if (cnt[a[r]] == 1) {
            cnt_zero--;
        }
        r++;
    }
}
```

Задача B. Тест на устойчивость

Для начала формализуем условие задачи. Дан массив из N положительных чисел. Требуется найти в нём подотрезок, сумма элементов которого равна M.

Рассмотрим простое решение. Переберём левую границу отрезка l. После этого начнём перебирать правую границу отрезка от l до N, насчитывая сумму чисел на отрезке. Если в какой-то момент сумма чисел стала равна M, мы нашли ответ. Данный алгоритм работает с асимптотикой $O(N^2)$ и получает неполный балл.

Теперь рассмотрим оптимальное решение. Воспользуемся методом двух указателей. Сначала поставим указатели l и r на первый элемент массива, также создадим переменную, в которой будем поддерживать сумму на отрезке $[l; r]$. Будем повторять следующие действия: сдвигаем указатель r на один элемент вправо, прибавляем к сумме значение добавленного элемента. Если сумма на отрезке меньше M, переходим к следующей итерации. Если сумма на отрезке равна M, мы нашли ответ. В противном случае будем сдвигать указатель l на один элемент вправо и вычитать значение удалённого элемента из суммы до тех пор, пока сумма на отрезке $[l; r]$ больше M. После этого, если сумма на отрезке равна M, мы нашли ответ.

Асимптотика: $O(N)$, так как оба указателя движутся только направо, а значит суммарно они сдвинутся не более $2 \cdot N$ раз.

Альтернативное решение. Насчитаем массив $pref$ таким образом, что $pref_i = \sum_{j=1}^{i} a_j$. Теперь мы можем с асимптотикой $O(1)$ находить сумму на отрезке $[l; r]$. Для этого нужно вычислить значение $pref_r - pref_{l-1}$. Переберём левую границу отрезка l. Теперь воспользуемся двоичным поиском. Пусть двоичный поиск зафиксировал правую границу отрезка r. Если сумма на отрезке $[l; r]$ меньше M, сдвинем левую границу бинарного поиска, в противном случае сдвинем правую границу бинарного поиска.

Асипмтотика: $O(N \log N)$.

Задача C. Шарики

Задача "три-в-ряд"очень просто сводится к обычному стеку. Для этого нам достаточно хранить последние два элемента стека. Каждый раз, когда мы пытаемся добавить следующий элемент, мы просто проверяем, равен ли он двум последним элементам в стеке. Если это так, то удаляем последние два элемента и все последующие выставленные шарики такого же цвета. Если же нет, то просто добавляем в стек этот шарик и по-прежнему поддерживаем два последних элемента на стеке. Таким образом, мы каждый шар добавим в стек не более одного раза, и удалим не более одного раза. Асимптотика $O(n)$.

Задача D. Город Че

Найдем для каждой точки число точек слева от нее на расстоянии больше R, это можно сделать двумя указателями.

Задача E. Дюбели и сверла

Задача была разобрана на лекции

```
int i = 0, j = 0, mn = INT_MAX;
while (i < n && j < m) {
    mn = min(abs(a[i] - b[j]), mn);
    if (i < n && a[i] < b[j]) {
        i++;
    } else {
        j++;
    }
}
```

Задача F. Стильная одежда

Сложим все массивы в один как пары (знаечение, номер массива) и отсортируем. Теперь нужно найти отрезок содержащий все четыре типа элементов, разница между первым и последним минимальна. Применим два указателя и найдем требуемый массив.

Задача G. Объединение последовательностей

Это задача на два указателя.

Задача H. Поездка на олимпиаду

Задача разбиралась на лекции, но теперь нужно не какой-то отрезок, а минимальной длины. Тогда будем двигать границу l если сумма больше или равна k.

```
while (r < n || l < n) {
    if (sum < k && r < n) {
        sum += a[r];
        r++;
    } else {
        if (sum == k && min_len > r - l) {
            // recalc answer
        }
        sum -= a[l];
        l++;
    }
}
```

Задача I. Плагиат

Отсортируем все файлы по возрастанию размера. Теперь решим задачу, используя метод двух указателей. Пусть сейчас указатели стоят в позициях l и r, тогда при перемещении указателя

r в позицию $r + 1$, указатель l может переместиться вправо, либо остаться на месте, так как $\frac{size(f_{r+1})}{size(f_l)} \geqslant \frac{size(f_r)}{size(f_l)}$. Для каждого положения указателей нужно прибавить к ответу $r - l$. Асимптотика: $\mathcal{O}(N \log N)$.

Задача J. Стильная одежда

Задача была разобрана на лекции

```
int i = 0, j = 0, mn = INT_MAX;
while (i < n && j < m) {
    mn = min(abs(a[i] - b[j]), mn);
    if (i < n && a[i] < b[j]) {
        i++;
    } else {
        j++;
    }
}
```