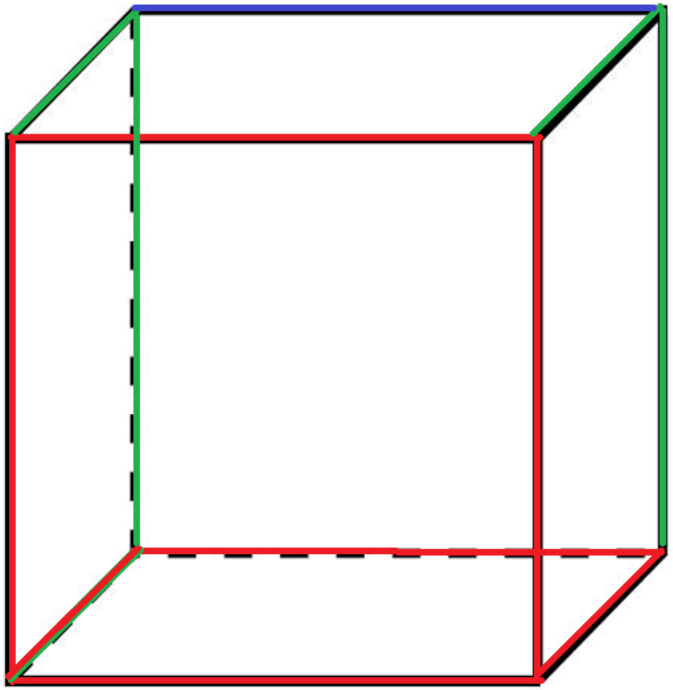
Муниципальный этап Всероссийской олимпиады школьников по информатике 9-11 класс

Разбор задач

Липецк, 2017

Задача А

Конструируем куб



Условие задачи

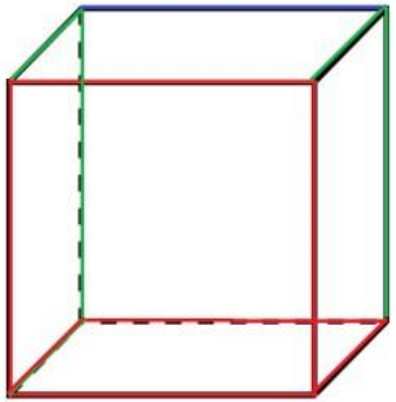
У Миши есть кусок проволоки длиной ***n*** метров. Миша хочет сделать из нее каркас куба со стороной ***a***. Для этого ему придется разрезать проволоку в нескольких местах. Разумеется, Миша, как умный ребенок хочет сделать как можно меньше разрезов куска проволоки. Помогите ему подсчитать минимальное количество разрезов.

В задаче нужно вывести YES, если проволоку можно разрезать на куски так, чтобы из нее можно было сконструировать каркас куба с ребром a. Во второй строке нужно вывести минимальное количество разрезов, которое нужно сделать Мише. Если из проволоки не удастся сконструировать куб, то выведете NO.

Как решать?

- Куб имеет 12 рёбер
- Общая длина рёбер равняется $12 \cdot a$
- Если $12 \cdot a > n$, то Мише не хватит проволоки, и ответ NO
- Если $12 \cdot a = n$, то Мише хватит проволоки, и ответ YES

Какое же минимальное количество разрезов нужно сделать Мише?
На рисунке мы видим четыре несвязных компоненты – четыре куска проволоки, из которых можно сложить куб.
То есть, потребуется 3 разреза.
3 разреза – минимальное количество.



Если 12a<n, то в этом случае остается лишняя часть провода в последнем кусе. И нам потребуется сделать еще один разрез, чтобы отрезать эту лишнюю часть. Ответ – 4 разреза.

- Сложность решения - $O(1)$

Задача В
Знакомимся с жителями

Условие задачи

Доктор Ватсон решил познакомиться со всеми жителями поместья Барскервилей. Число жителей в поместье нечетно и составляет n человек. Число жителей может быть достаточно велико, поэтому Ватсон решил ограничиться знакомством с тремя представителями поместья: **с самым бедным жителем, с жителем, обладающим средним достатком, и с самым богатым жителем.**

Согласно традициям поместья Баскервилей, считается, что житель обладает средним достатком, если при сортировке жителей по сумме денежных сбережений он оказывается ровно посередине. Известно, что каждый житель поместья имеет уникальный идентификационный номер, значение которого расположено в границах от единицы до n.

Информация о размере денежных накоплений жителей хранится в массиве таким образом, что сумма денежных накоплений жителя, обладающего идентификационным номером ***i***, содержится в ячейке с номером ***i*** . Помогите доктору Мортимеру, который готовит встречу Ватсона с жителями, вычислить идентификационные номера жителей, которые будут приглашены на встречу с Ватсоном.

Как решать?

- Пусть денежные накопления жителей хранятся в массиве $a[i]$
- Отсортируем по первому значению массив, состоящий из пар вида $(a[i], i)$
- После сортировки выведем вторые значения пар элементов с индексами: $0, \frac{n}{2}, n-1$ (индексация с нуля)
- Сложность решения - $O(nlogn)$

```
double a[N];
pair<double, int> b[N];
// Считывание данных...
for (int i = 0; i < n; ++i) {
     b[i] = make_pair(a[i], i);
}
sort(b, b + n);
cout << b[0].second + 1 << ' ' << b[n / 2].second + 1 << ' ' << b[n - 1].second + 1;
```

Задача С
Играем в игры

Условие задачи

Задана последовательность ***a***, состоящая из ***n*** целых чисел. Игрок может сделать несколько ходов. За один ход игрок может выбрать некоторый элемент последовательности (обозначим выбранный элемент a_k) и удалить его, при этом из последовательности также удаляются все элементы, равные $a_k + 1$ и $a_k - 1$. Описанный ход приносит игроку a_k очков.

Максим, как и все подростки - максималист и поэтому хочет набрать как можно больше очков. Какое максимальное количество очков он сможет набрать?

Рассмотрим тестовый пример. N=9
{1 2 1 3 2 2 2 2 3}

В этом примере предлагаем Максиму такую последовательность действий. Первоначально надо выбрать любой элемент, равный 2. В этом случае из последовательности исчезнут элементы, равные 1 и 3, а Максим заработает 2 балла. Последовательность станет равна [2, 2, 2, 2]. Далее Максим сделает еще 4 хода, на каждом ходу выберет любой элемент, равный 2. Итого Максим заработает 10 очков. Больше 10 баллов Максим заработать не сможет, какие бы другие ходы он не делал.

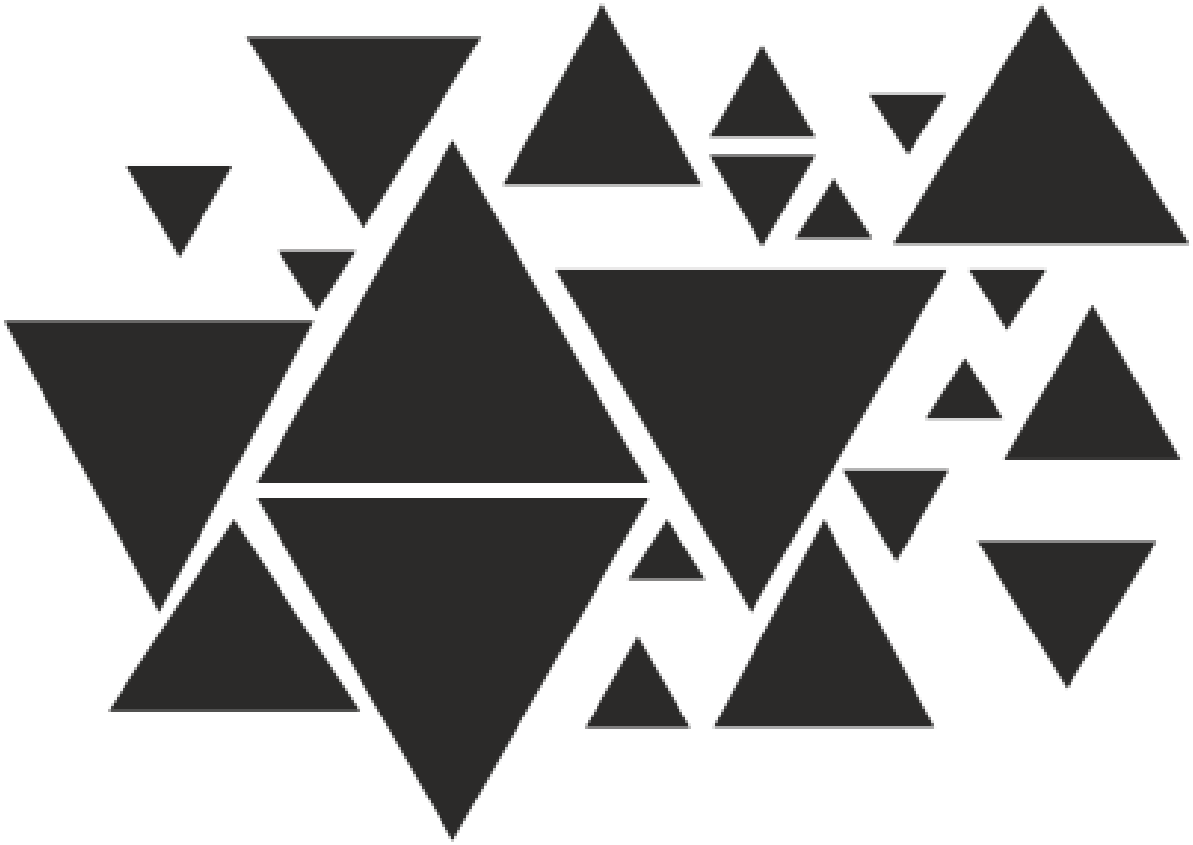
Как решать?

- Для начала заполним массив cnt, где $cnt[i]$ равно количеству вхождений числа i в последовательность a
- Далее воспользуемся динамическим программированием: $dp[i]$ - максимальное количество набранных очков, если рассмотрены числа, меньшие либо равные i И некоторые из них взяты.
- Инициализация: $dp[0] = 0, dp[1] = cnt[1]$ Других чисел нет.
- Переход: $dp[i] = \max(dp[i-1], dp[i-2] + cnt[i] \cdot i)$
- Ответ будет лежать в ячейке $dp[MAXN]$, где $MAXN$ – максимальное число в последовательности a
- Сложность решения - $O(n + MAXN)$

Как решать?

```
const int MAXN = 100100;
int a[MAXN], cnt[MAXN], dp[MAXN];
// Считывание данных...
for (int i = 0; i < n; ++i) {
    ++cnt[a[i]];
}
dp[0] = 0;
dp[1] = cnt[1];
for (int i = 2; i < MAXN; ++i) {
    dp[i] = max(dp[i - 1], dp[i - 2] + cnt[i] * i);
}
cout << dp[MAXN - 1];
```

Задача D
Масштабируем треугольники



Условие задачи

Максим вдоволь наигрался с числами и в данный момент изучает треугольники, которые подходят для геометрической игры. Он начинает с равностороннего треугольника со сторонами, равными ***x***, и ему необходимо выполнить несколько операций так, чтобы получить равносторонний треугольник с меньшими сторонами, равными ***y***.

По правилам геометрической игры за одну секунду Максим меняет длину ровно одной стороны текущего треугольника так, чтобы он оставался невырожденным треугольником (то есть треугольником ненулевой площади). В любой момент времени все стороны треугольника должны иметь целочисленную длину.

За какое минимальное количество секунд Максим сможет получить равносторонний треугольник с длиной стороны ***y***?

Как решать?

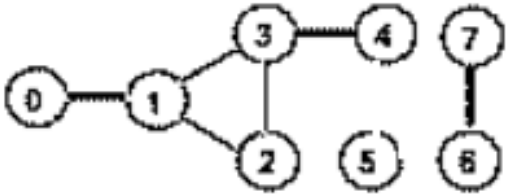
- Для удобства развернём процесс: получим из треугольника со стороной y треугольник со стороной x
- Воспользуемся жадным алгоритмом: будем увеличивать меньшую сторону настолько, насколько это возможно, чтобы треугольник оставался невырожденным
- Будем повторять это действие, пока меньшая сторона меньше, чем x
- Очевидно, что если какие-то стороны треугольника оказались больше x, то мы могли получить стороны длины ровно x.
- Сложность решения - $O(logC)$, где $C = y - x$

```
int a = y, b = y, c = y;
int answer = 0;
while (a < x) {
    ++answer;
    a = b;
    b = c;
    c = a + b - 1;
`
```

Задача E
Ищем уязвимые места в сети

Условие задачи

В компьютерной сети соединение L двух серверов, считается уязвимым соединением сети, если есть как минимум два сервера, таких, что все пути сетевого взаимодействия между A и B проходят через соединение L. На рисунке изображена компьютерная сеть с тремя уязвимыми соединениями: 0-1, 3-4, 7-6



Искать уязвимые места вручную – довольно сложное и трудоемкое занятие. Поэтому вас просят написать программу, которая по описанию структуры компьютерной сети вывела бы все уязвимые соединения в ней.

Как решать?

- Формально говоря, в данной задаче дан неориентированный граф, в котором необходимо найти все мосты
- Напомним, что мостом называется ребро, после удаления которого граф распадается на несколько компонент связности
- Очевидно, что если граф изначально не связный, то задачу можно решать отдельно для каждой компоненты связности
- Для неполного решения было достаточно перебрать ребро в графе и проверить, является ли оно мостом (перестроить граф и запустить обход в глубину)
- Сложность такого решения - $O(m^2 + n \cdot m)$
- Для полного решения необходим алгоритм, основанный на поиске в глубину

Как решать? Полное решение

```
int tin[N], go[N];
bool used[N];
vector<pair<int, int>> g[N];
vector<int> bridges;
int timer = 0;
void dfs(int v, int p = -1) {
    used[v] = true;
    tin[v] = timer++;
    go[v] = tin[v];
    for (int i = 0; i < g[v].size(); ++i) {
        int to = g[v][i].first, id = g[v][i].second;
        if (to == p) {
            continue;
        }
        if (!used[to]) {
            dfs(to, v);
            go[v] = min(go[v], go[to]);
            if (go[to] > tin[v]) {
                bridges.push_back(id);
            }
        } else {
            go[v] = min(go[v], tin[to]);
        }
    }
}
```

Спасибо за внимание! Вопросы?