
Факультет программной инженерии и компьютерной техники

Алгоритмы и структуры данныхка

# Yandex Contest

## Введение в алгоритмы

Преподаватель: Косяков Михаил Сергеевич
Выполнил: Ле Чонг Дат
Группа: P3231

2021 г.

Часть I

# E. Коровы в стойла

## 1 Решение

- Если на расстоянии $d$ мы не можем расположить коров так, чтобы минимальное расстояние между ними было $\geq d$, то, очевидно, мы не можем сделать это с помощью $d+1 \rightarrow$ Мы можем выполнить двоичный поиск результатов.
- Предположим, что $k$ киосков выбраны как $i_1, i_2, ..., i_k$ соответственно. Очевидно, $i_1 = 1$, потому что если $i_1 \neq 1$, мы все равно можем заменить $i_1$ на 1, потому что, очевидно, расстояние от первого стойла до $stall_{i_2}$ будет больше, чем расстояние от $stall_{i_1}$ до $stall_{i_2}$. Мы проверим, существует он или нет, выбрав $k$ stall to distance $\geq d$ или нет. При $i_1 = 1$, очевидно, аналогично $i_1$, $i_2$ должен быть ближайшим к $i_1$ стойлом, удовлетворяющим расстоянию между двумя стойлами $i_1$ и $i_2 \geq d$. Подобно $i_3$, $i_4$, ..., если $i_k$ найден, то $d$ удовлетворяется, и наоборот.

## 2 Реализация

```
#include <bits/stdc++.h>
using namespace std;
const int N = 1e5 + 10;
int n, k;
int stalls[N];
bool valid(int mid) {
    int cur = stalls[1], cnt = 1;
    for(int i = 2; i <= n; ++i)
        if (stalls[i] - cur >= mid) {
            cnt ++;
            cur = stalls[i];
        }
    return (cnt >= k);
}
int main() {
    ios::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);

    cin >> n >> k;
    for(int i = 1; i <= n; ++i)
        cin >> stalls[i];
    int lo = 0, hi = 1e9, ans = 0;
    while (lo <= hi) {
        int mid = (lo + hi) >> 1;
        if (valid(mid)) {
            ans = mid;
            lo = mid + 1;
```

```
        } else hi = mid - 1;
    }
    cout << ans;
}
```

# Часть II

# F. Число

## 1 Решение

$A = a_1a_2a_3...a_n$; $B = b_1b_2b_3...b_m$
Без ограничения общности предположим, что n $\leq$ m.

- Случай 1: A = B $\rightarrow$ A идет первым B или B идет первым A не имеет значения.
- Случай 2: A - префикс B. Если $b_{n+1}b_{n+2}...b_m > a_1a_2...a_n$, то B перед A даст результат больше .
- Случай 3: A не является префиксом B. Если A > B находится в словарном порядке, A идет первым, и наоборот.

## 2 Реализация

```
#include <bits/stdc++.h>
using namespace std;
vector<string> nums;
string s;
bool compare(string a, string b) {
    for(int i = 0; i < min(a.length(), b.length()); ++i) {
        if (a[i] != b[i]) return a[i] > b[i];
    }
    if (a.length() == b.length()) return false;
    return a.length() > b.length() ? compare(a.substr(b.length(), a.length()), b) :
compare(a, b.substr(a.length(), b.length()));
}
int main() {
    ios::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);
    while (cin >> s) nums.emplace_back(s);
    sort(nums.begin(), nums.end(), compare);
    for(auto num: nums) cout << num;
}
```

Часть III

# G.Кошмар в замке

## 1 Решение

Буква самого высокого значения должна помещать 1 букву в начало и 1 букву в конец строки. Вторая по величине буква значения должна помещать 1 букву в следующее начало и 1 букву перед концом строки, 3-е наивысшее значение, 4-е наивысшее значение, ... Остальные буквы будут помещены в середину строки нить.

## 2 Реализация

```
#include <bits/stdc++.h>
using namespace std;
string s, head, tail;
int cnt[30];
int main() {
    ios::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);

    cin >> s;
    vector<tuple<long long, int>> costs;
    for(int i = 0; i < 26; ++i) {
        long long cost;
        cin >> cost;
        costs.emplace_back(cost, i);
    }
    for(auto c: s) cnt[c - 'a'] ++;
    sort(costs.begin(), costs.end(), greater<tuple<long long, int>>());
    for(auto cost: costs) {
        long long val;
        int ch;
        tie(val, ch) = cost;
        if (cnt[ch] < 2) continue;
        cnt[ch] -= 2;
        head += char(ch + 'a');
        tail += char(ch + 'a');
    }
    reverse(tail.begin(), tail.end());
    for(int i = 0; i < 26; ++i)
        while (cnt[i]) {
            head += char(i + 'a');
            cnt[i] --;
        }
    cout << head + tail;
```

```
}
```

# Часть IV
# H. Магазин

## 1 Решение

Расположите достоинства монет в порядке возрастания. $a_1, a_2, ..., a_n$. Наибольшая удаляемая сумма будет $a_{n+1-k} + a_{n+1-2*k} + a_{n+1-3*k} + ...+$

## 2 Реализация

```
#include <bits/stdc++.h>
using namespace std;
const int N = 1e5 +1;
int n, k;
int a[N];
int main() {
    ios::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);

    cin >> n >> k;
    long long tot = 0;
    for(int i = 0; i < n; ++i) {
        cin >> a[i];
        tot += a[i];
    }
    sort(a, a + n);
    for(int i = n - k; i >= 0; i -= k)
        tot -= a[i];
    cout << tot;
}
```

# Часть V
# Вывод

Проблемы на практике с использованием алгоритма сортировки являются обычными. Я научился использовать сортировку для решения механических проблем