# JakeMate14

## Contents



# 1 Estructuras de Datos

## 1.1 Segment tree

```
struct segtree {
    int size;
    vector<ll> vv;

    void build(vector<int> &nums) {
        size = 1;
        while (size < nums.size()) size *= 2;
        vv.assign(2 * size, 0);
        build(nums, 0, 0, size);
    }

    void build(vector<int> &nums, int x, int lx, int rx) {
        if (rx - lx == 1) {
            if (lx < nums.size()) {
                vv[x] = nums[lx];
            }
        } else {
            int m = (lx + rx) / 2;
            build(nums, 2 * x + 1, lx, m);
            build(nums, 2 * x + 2, m, rx);
            vv[x] = vv[2 * x + 1] + vv[2 * x + 2];
        }
    }

    void set(int i, int v) {
        set(i, v, 0, 0, size);
    }

    void set(int i, int v, int x, int lx, int rx) {
        if (rx - lx == 1) {
            vv[x] = v;
        } else {
            int m = (lx + rx) / 2;
            if (i < m) {
                set(i, v, 2*x+1, lx, m);
            } else {
                set(i, v, 2*x+2, m, rx);
            }
            vv[x] = vv[2*x+1] + vv[2*x+2];
```

```
        }
    }

    ll sum(int l, int r) {
        return sum(l, r, 0, 0, size);
    }

    ll sum(int l, int r, int x, int lx, int rx) {
        if (r <= lx) return 0;
        if (l >= rx) return 0;
        if (lx >= l && rx <= r) return vv[x];
        int m = (lx + rx) / 2;
        ll s1 = sum(l, r, 2*x+1, lx, m);
        ll s2 = sum(l, r, 2*x+2, m, rx);
        return s1 + s2;
    }
};
```