树状数组

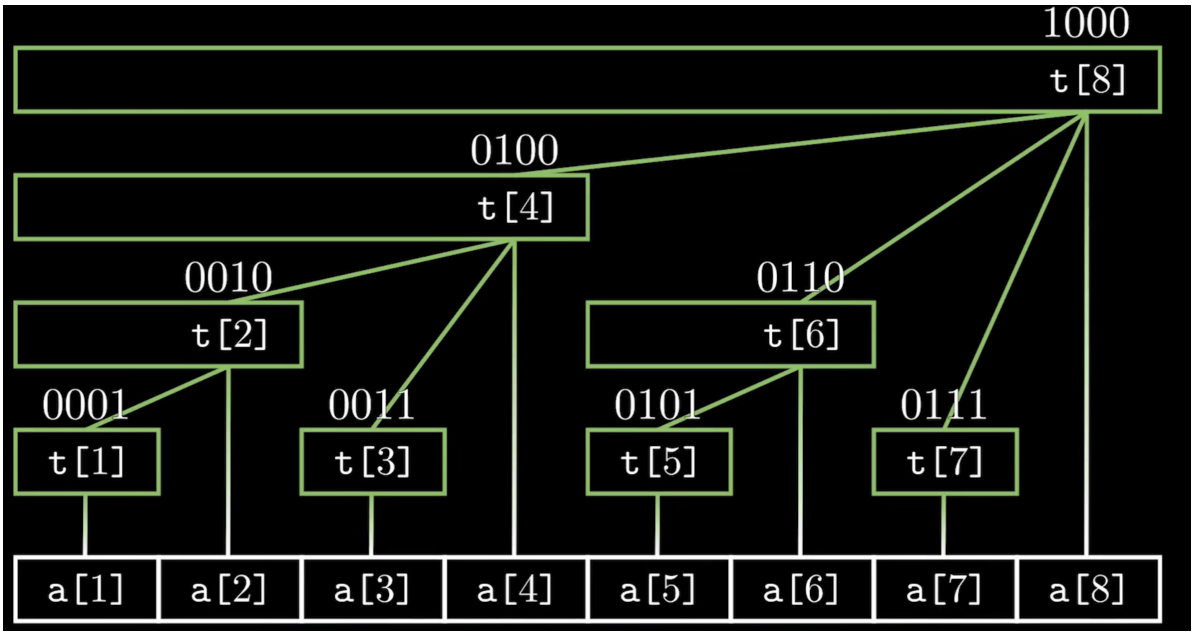
算法作用

- 动态维护前缀和、异或和、最大值、最小值
- 单点修改|查询前缀和、单点修改|单点查询、单点修改|区间查询、区间修改|单点查询
- **无法维护最大值**

前置知识

lowbit(x)运算：x&-x

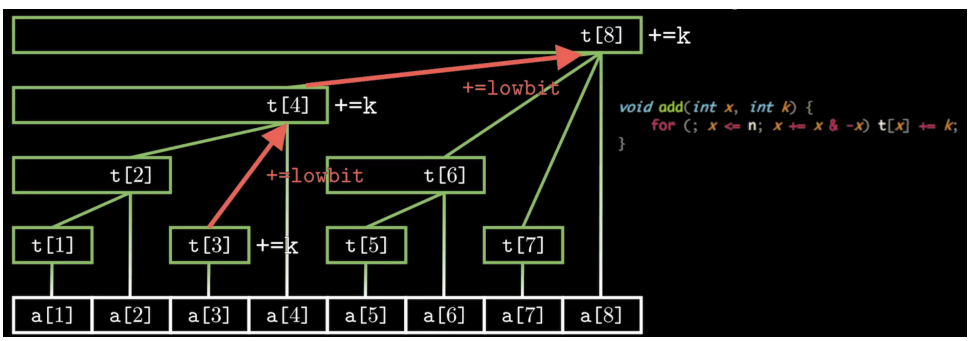
一些公式



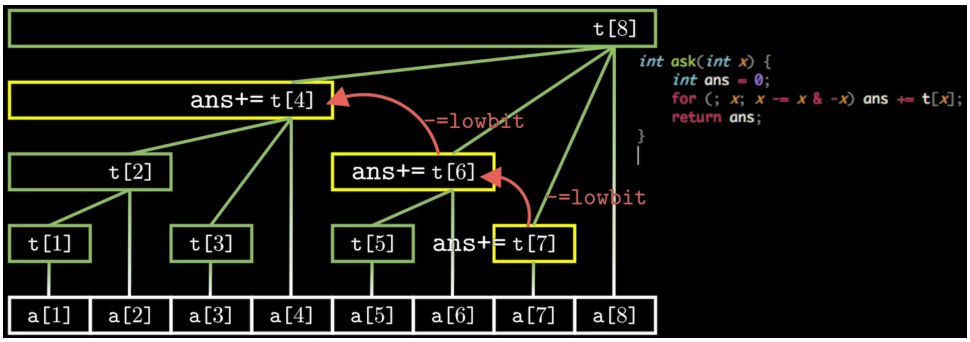
对于任意一个数x:

- $t[x]$保存以x为根的子树中叶节点的**和**.
- $t[x]$节点所覆盖的长度等于$lowbit(x)$
- $t[x]$节点的父节点为$t[x+lowbit(x)]$
- $t[x]$节点的左侧相邻节点为$t[x-lowbit(x)]$
- 整个树的深度是$log_2(n+1)$

操作

- **add(x, k)** — 对a[x] 加上一个数k
  - 

- 依次向上修改父节点(`idx += lowbit(idx)`)，并对所有父节点进行更新
- ```
  void add(int x, int k){
      for(; x <= n; x += lowbit(x)) t[x] += k;
  }
  ```

- **ask(x)** — 查询从a[1] 到 a[x] 区间的前缀和
  - 

  - 从该点向左上找到与其相邻的节点(`idx -= lowbit(idx)`)
  - ```
    int ans(int x){
        int ans = 0;
        for(; x; x -= lowbit(x)) ans += t[x];
        return ans;
    }
    ```

- 单点修改|查询前缀和

  ```
  add(x, k);
  ask(k);
  ```

- 单点修改|单点查询

  ```
  add(x, k);
  ask(x) - ask(x - 1);
  ```

- 单点修改|区间查询

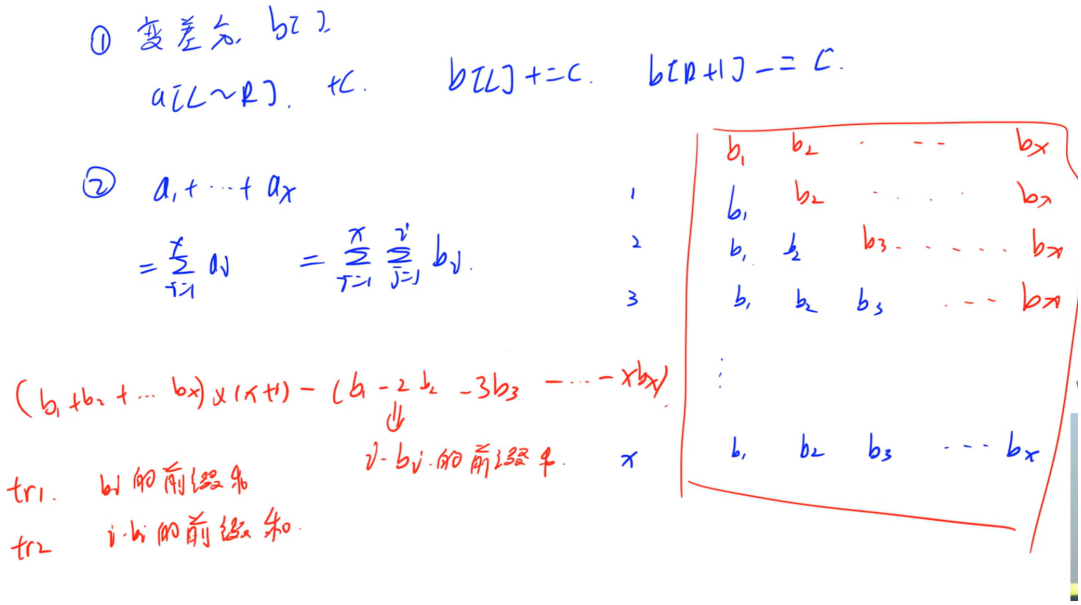
  ```
  add(x, k);
  ask(r) - ask(l - 1);
  ```

- 区间修改|单点查询 **差分**

  注意：需要用树状数组维护原数组的差分数组的前缀和

  ```
  add(l, d), add(r + 1, -d);
  a[x] + ask(x);
  ```

- 区间修改|区间查询



```
//初始化
add(tr1, i, b);
add(tr2, i, i * b);
//添加
add(l, d, tr1), add(r + 1, -d, tr1);
add(l, l * d, tr2), add(r + 1, (r + 1) * -d, tr2);
//查询
int get(int x){
    return ask(x, tr1) * (x + 1) - ask(x, tr2);
}
get(r) - get(l - 1)
```

例题

P3374 【模板】树状数组 1

```
#include <bits/stdc++.h>
#define endl "\n"
#define int long long

using namespace std;

const int maxn = 5e5 + 10;
int tr[maxn];
int a[maxn];
int n, m;

int lowbit(int x){
    return x & -x;
}

void add(int x, int b){
    for(; x <= n; x += lowbit(x)) tr[x] += b;
}

int ask(int x){
```

```
    int ans = 0;
    for(; x; x -= lowbit(x)) ans += tr[x];
    return ans;
}

signed main(){
    ios::sync_with_stdio(false); cin.tie(0); cout.tie(0);

    cin >> n >> m;
    for(int i = 1; i <= n; i ++){
        cin >> a[i];
        add(i, a[i]);
    }
    while(m -- ){
        int op; cin >> op;
        if(op == 1){
            int a, b; cin >> a >> b;
            add(a, b);
        }
        else{
            int a, b; cin >> a >> b;
            cout << ask(b) - ask(a - 1) << endl;
        }
    }


    return 0;
}
```

P3368 【模板】树状数组 2

```
#include <bits/stdc++.h>
#define endl "\n"
#define int long long
using namespace std;

const int maxn = 5e5 + 10;
int n, m;
int tr[maxn], a[maxn];

int lowbit(int x){
    return x & -x;
}

void add(int x, int k){
    for(; x <= n; x += lowbit(x)) tr[x] += k;
}

int ask(int x){
    int ans = 0;
    for(; x; x -= lowbit(x)) ans += tr[x];
    return ans;
}

signed main(){
    ios::sync_with_stdio(false); cin.tie(0); cout.tie(0);
    cin >> n >> m;
```

```
    for(int i = 1; i <= n; i ++){
        cin >> a[i];
        add(i, a[i] - a[i - 1]);
    }

    while(m--){
        int op; cin >> op;
        if(op == 1){
            int x, y, k; cin >> x >> y >> k;
            add(x, k), add(y + 1, -k);
        }
        else{
            int k; cin >> k;
            cout << ask(k) << endl;
        }
    }


    return 0;
}
```

AcWing 241. 楼兰图腾

```
#include <bits/stdc++.h>
#define endl "\n"
#define int long long
using namespace std;

const int maxn = 2e5 + 10;
int n, m;
int ans1, ans2;
int a[maxn], upper[maxn], lower[maxn];
int tr[maxn];

int lowbit(int x){
    return x & -x;
}

void add(int a, int b){
    for(; a <= n; a += lowbit(a)) tr[a] += b;
}

int ask(int x){
    int ans = 0;
    for(; x; x -= lowbit(x)) ans += tr[x];
    return ans;
}

signed main(){
    ios::sync_with_stdio(false); cin.tie(0); cout.tie(0);
    cin >> n;
    for(int i = 1; i <= n; i ++) cin >> a[i];

    for(int i = 1; i <= n; i ++){
        int y = a[i];
        lower[i] = ask(y - 1);
        upper[i] = ask(n) - ask(y);
```

```
35
36          add(y, 1);
37      }
38
39      memset(tr, 0, sizeof tr);
40
41      for(int i = n; i >= 1; i --){
42          int y = a[i];
43          ans2 += ask(y - 1) * lower[i];
44          ans1 += (ask(n) - ask(y)) * upper[i];
45          add(y, 1);
46      }
47
48      cout << ans1 << " " << ans2;
49
50      return 0;
51  }
```