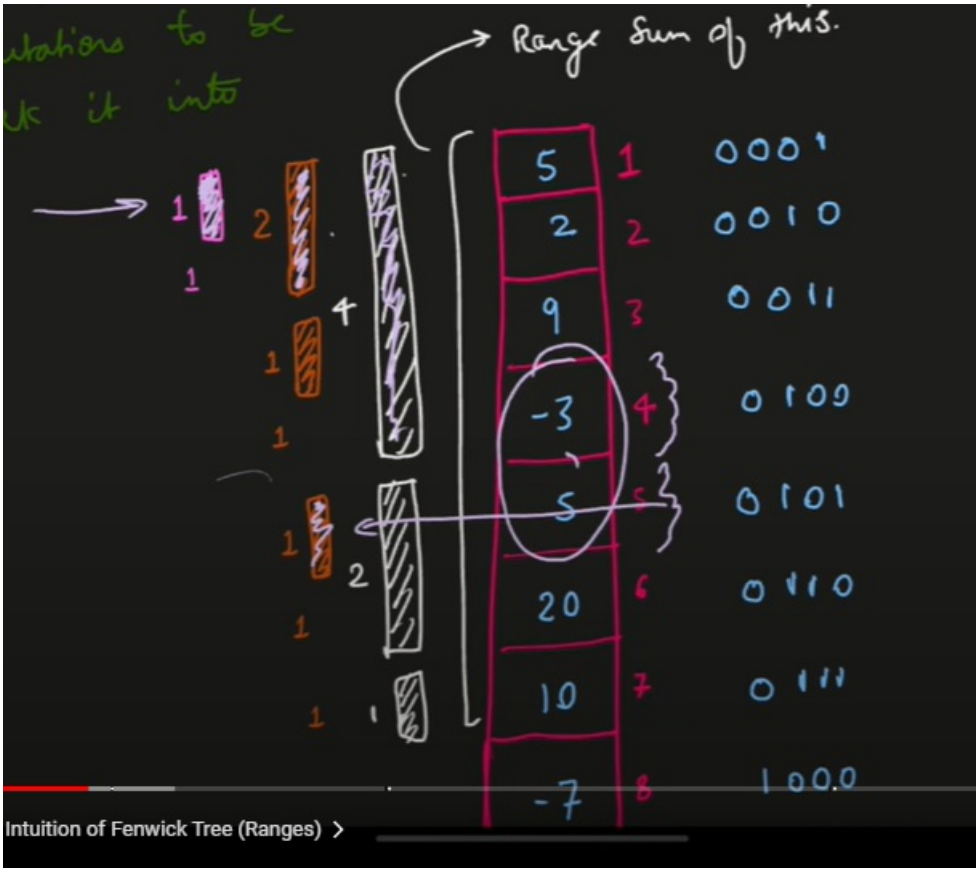
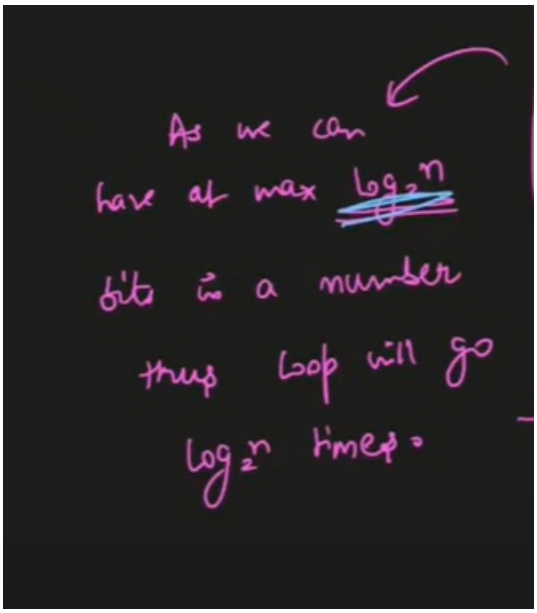
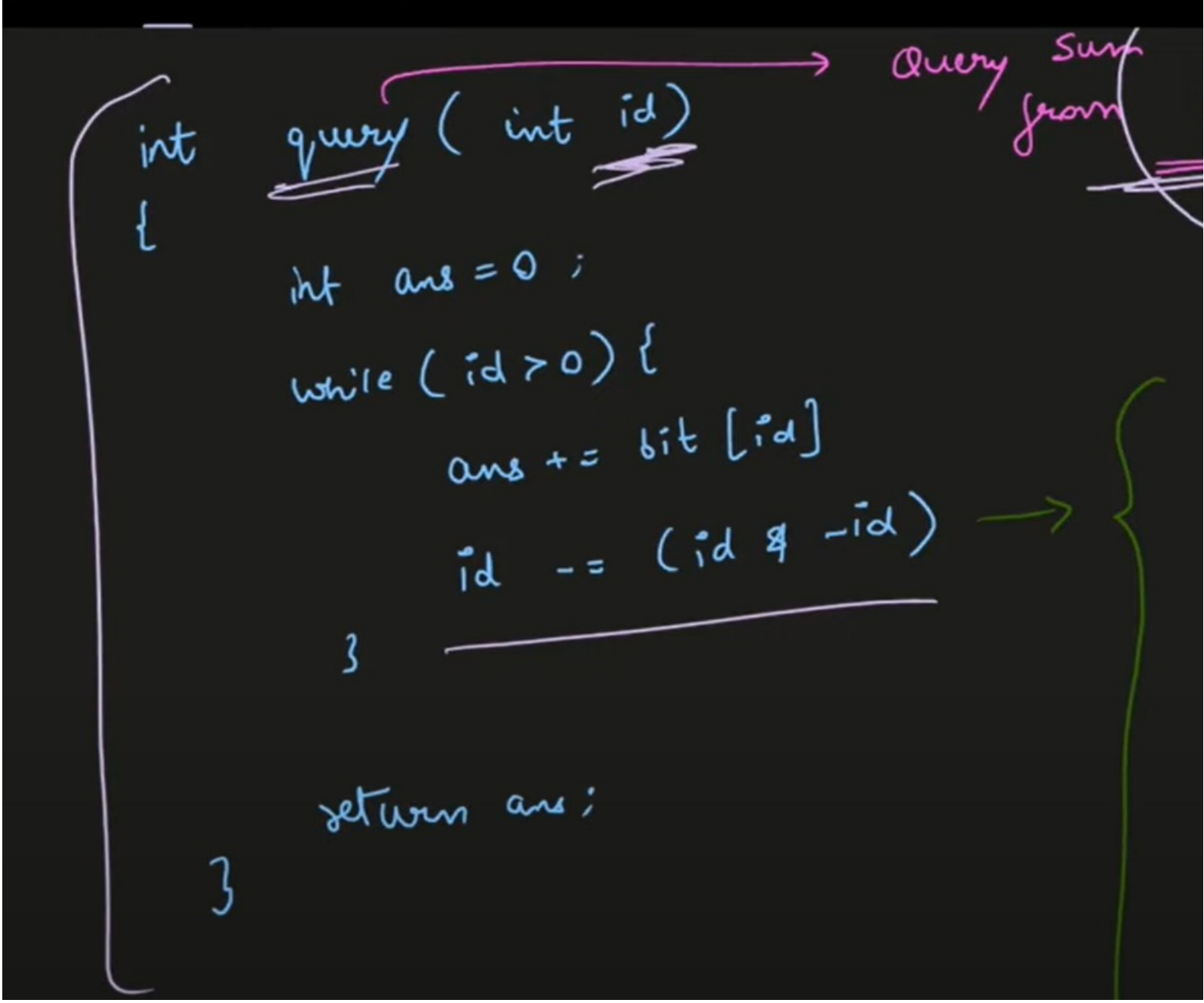
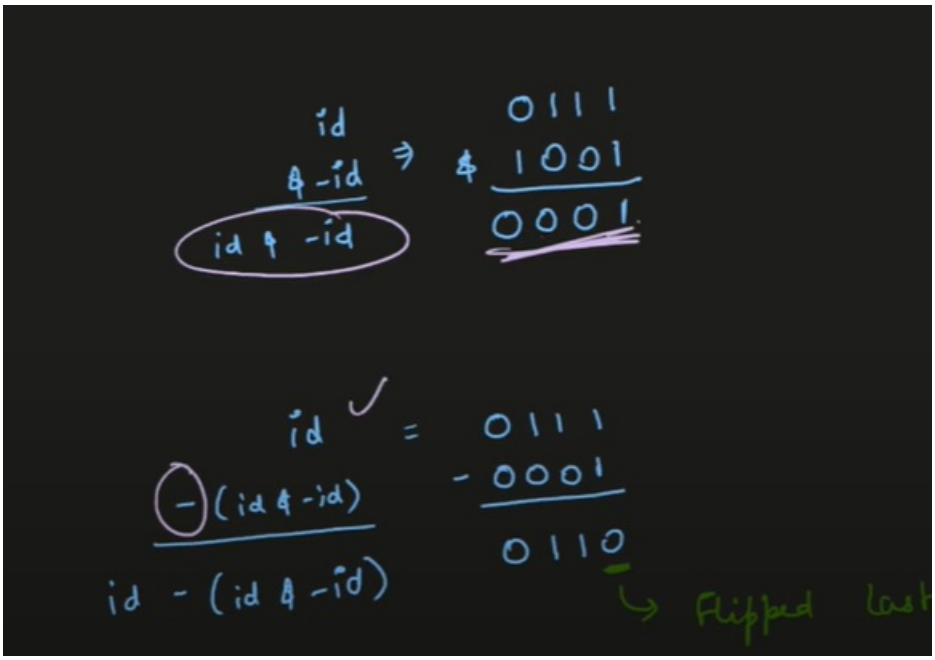

Range sum of this.
5
2
9
-3
5
20
10
-7
1
2
3
4
5
6
7
8
0001
0010
0011
0100
0101
0110
0111
1000
Intuition of Fenwick Tree (Ranges) >

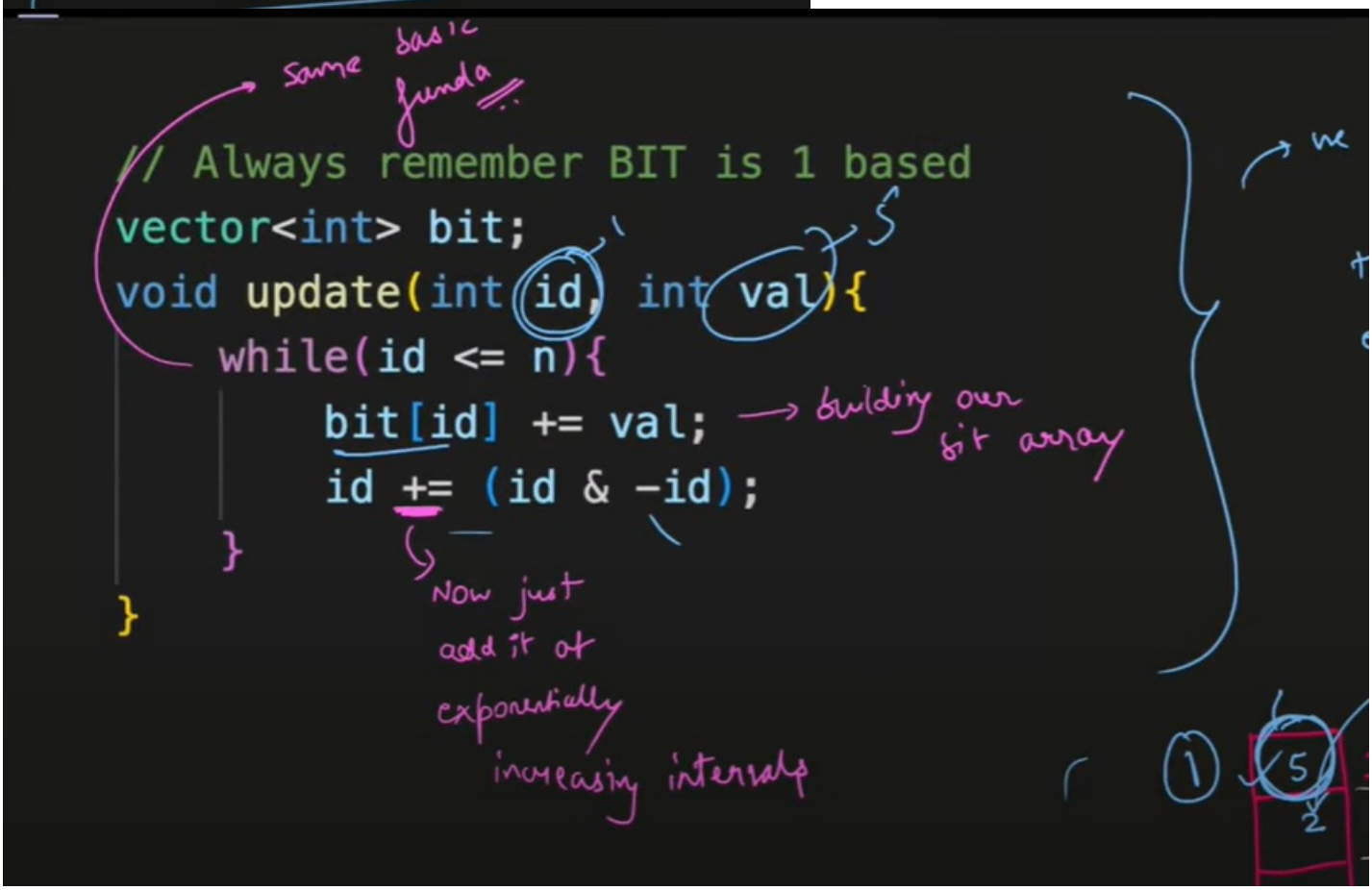

As we can have at max $\log_2 n$ bits in a number thus loop will go $\log_2 n$ times.





So just rightmost set bit is flipped

```
int query(int id){
    int ans = 0;
    while(id > 0){
        ans += bit[id];
        id -= (id & -id);
    }
    return ans;
}
```



vector<int> bit; // BIT array

vector<int> a; // Original input array

void update(int id, int val) {

while (id < bit.size()) {

bit[id] += val;

id += (id & -id);

}

}

int query(int id) {

int ans = 0;

while (id > 0) {

ans += bit[id];

id -= (id & -id);

}

```
return ans;

}

int main() {

int n, q;

cin >> n;

bit = vector<int>(n + 1, 0);

a = vector<int>(n + 1);

for (int i = 1; i <= n; ++i) {

cin >> a[i];

update(i, a[i]);

}

cin >> q;

while (q--) {

int typeOfQuery;

cin >> typeOfQuery;

if (typeOfQuery == 1) {

int L, R;

cin >> L >> R;

int ans = query(R) - query(L - 1);

cout << ans << endl;

} else {

int id, val;

cin >> id >> val;

update(id, -a[id]); // Firstly make it back to 0

a[id] = val; // Update your original input array 'a'

update(id, a[id]); // Update the BIT array

}

}

return 0;

}
```

https://leetcode.com/problems/range-sum-query-mutable/