Seminarska naloga 2

8.2 Intervalnih drevesa

Intervalnih drevesa (angl. segment trees) so podatkovna struktura, ki lahko učinkovito odgovori na dinamično poizvedbe o obsegu (angl. range queries).

Primer takih poizvedb o obsegu je težava pri iskanju najmanjšega indeksa elementa v matriki [i..j] (angl. Range Minimum Query (RMQ)).

Array	Values	18	17	13	19	15	11	20
A	Indices	0	1	2	3	4	5	6

Slika 8.7: RMQ problem.

RMQ(3,4) = 4

RMQ(0,0) = 0

RMQ(0,1) = 1

RMQ(0,6) = 5

Obstaja več načinov za izvajanje RMQ. En trivialni algoritem je preprosto ponovitev matrika od indeksa i do j in poroča indeks z najmanjšo vrednostjo, vendar se bo ta pognal v O(n) času. Problem: če je n velik in obstaja veliko poizvedb.

Intervalno drevo razporedi podatke v binarnem drevesu.

Indeks 1 (preskok indeksa 0) je korens in levi in desni otrok indeksa p sta indeksi $2 \times p$ in $(2 \times p) + 1$.

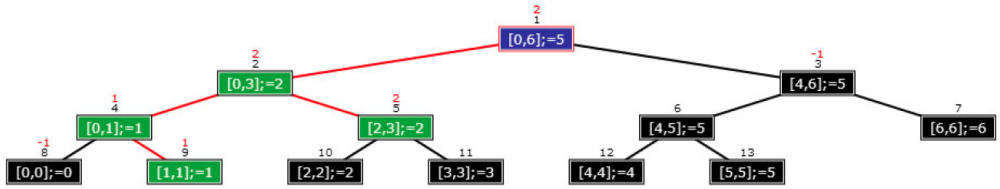
Vrednost $st[p]$ je vrednost RMQ v obsegu, povezanega z indeksom p.

Koren intervalnega drevesa predstavlja segment $[0, n-1]$. Za vsak shranjeni segment [L, R] v indeksu p, kjer je L! = R, bo obseg razdeljen na $[L, (L+R)/2]$ in $[(L+R)/2+1, R]$ v levem in desnem vozlišču. Levi podsegment in desni podsegment bosta shranjena v indeksu $2 \times p$ in $(2 \times p) + 1$. Ko je L = R, je jasno, da je $st[p] = L$ (ali R). Intervalno drevo bomo rekurzivno zgradili, pri čemer bomo primerjali najmanjšo vrednost leve in desne podsegmenti in posodabljati

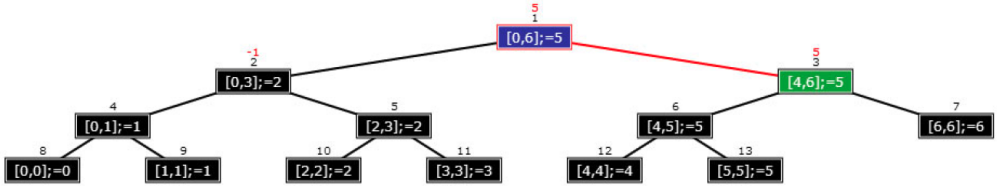
$st[p]$ segmenta.

Ustvarajanje intervalnega drevesa je O(2n).

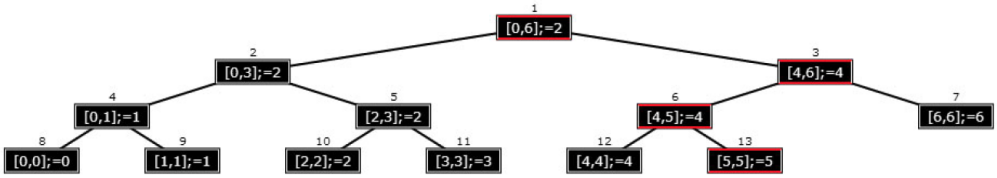
Ko je intrvslno drevo pripravljeno, lahko odgovor na RMQ opravite v O (log n). Odgovor za RMQ (i, i) je trivialno - preprosto vrni sebe. Za splošni primer pa RMQ (i, j), potrebni so nadaljnji pregledi. $p_1 = RMQ(i, (i + j)/2)$ in $p_2 = RMQ((i + j)/2 + 1, j)$. Potem je RMQ (i, j) p_1, če je $A[p1]A[p2]$ ali p_2 drugače.



Slika 8.8: Intervalno drevo od A = {18, 17, 13, 19, 15, 11, 20} and RMQ(1, 3).



Slika 8.9: Intervalno drevo od A = {18, 17, 13, 19, 15, 11, 20} and RMQ(4, 6).



Slika 8.10: Posodabljanje A = {18, 17, 13, 19, 15, **99**, 20}.

```
class SegmentTree{
private: vi st,A;
int n;
int left (int p) {return p << 1;}
int right (int p) {return (p << 1) + 1;}

void build(int p, int L, int R){
    if (L == R){
        st[p] = L;
    }
    else {
        build(left(p),L, (L+R)/2);
        build(right(p),(L+R)/2+1,R);
```

```
        int p1 = st[left(p)], p2 = st[right(p)];
        st[p] = (A[p1] <= A[p2]) ? p1 :p2;
    }
}


int rmq(int p, int L, int R, int i, int j){
    if(i > R || j < L) return -1;
    if(L >= i && R <= j) return st[p];

    int p1 = rmq(left(p), L,(L+R)/2, i, j)
    int p2 = rmq(right(p),(L+R)/2+1, R, i, j)

    if (p1 == -1) return p2;
    if (p2 == -1) return p1;
    return (A[p1] <= A[p2]) ? p1 : p2;
}

public:
SegmentTree(const vi & A){
    A = _A; n= (int)A.size();
    st.assign(4*n,0);
    build(1,0,n-1);
}
};
int main(){
    int arr[] = {18, 17, 13, 19, 15, 11, 20};
    vi A(arr, arr+7);
    SegmentTree st(A);
    printf("RMQ(1, 3) = %d\n", st.rmq(1, 3));
    printf("RMQ(4, 6) = %d\n", st.rmq(4, 6));

}
```

Naloga

V starodavni gusarski dobi je bila Gusarska dežela razdeljena na dve skupini gusarjev, in sicer X in Y. Vsaka ekipa gusarjev ni bila določena. Pirati so napadli in zaplenjengea gusarja so odpeljali v drugo gusarsko ekipo. Kar naenkrat se je v Gusarski deželi pojavil čarovnik, kjer je po lastni volji opravljal prehod piratov iz ene ekipe v drugo ekipo. Proces menjave ekipe je bil znan kot mutiranje. Bilo je N piratov in vsi pirati imajo id od 0 do $N-1$. Čarovnik bi lahko mutiral kup piratov z zaporednimi ID-ji na drugega. Recimo, da je bilo v gusarski deželi 100 piratov in vsi so bili v ekipi Y. Čarovnik bi lahko naredil spremembo piratov z id-jem z 10 na 33 v X ekipe. V tem primeru, gusarska dežela bi imela 24 gusarjev v X in 76 v Y. Čarovnik je zelo hitro delal spremembe. To se pa Bogu ni dopadlo. Bog je imel naklonjenost ekipe X in je

vprašal čarovniku: »Koliko gusarjev so v ekipi X od indeksa 2 do 30? ". Zdaj je bil čarovnik zmeden, saj je bil učinkovit le pri spremebi, ne v štetju :-) Čarovnik je bil dovolj pameten, da je ujel pametnega človeka z Zemlje. In na žalost to si ti! Zdaj morate odgovoriti na vprašanja Bogu. :)

Prva vrstica vnosa bo vsebovala število testnih primerov T.

Za vsak testni primer:

Prvi del opisa bo gusarske dežele. Lahko bi bilo največ N (1≤N≤ 1024000) gusarejv. Vsakemu gusarju je dodeljen X ali Y ekipe. Gusari iz ekipe X oznacimo z '1' in gusari iz ekipe Y z '0'. Morate zgraditi niz gusarjev. Vsak primer se začne s celim številom M (M≤100), kjer sledijo vrstice M par. V vsakem paru vrstice (imenujemo jih niz), prvo ima celo število T (T≤200), naslednji pa niz gusarjev (sestavljen iz 0 in 1, 0 za X, 1 za Y, največja dolžina je 50). Za vsak par povežite niz gusarjev, T krat. Povežite vse nastale M sklope nizov, da zgradite niz guarjev v deželi. Končni sestavljeni niz opisuje gusarejv od indeksa 0 do konca $N-1$ (za N gusarjev).

Naslednji del vnosa vseboval poizvedbe. V prvi vrstici naslednjega dela je celo število Q, ki opisuje število poizvedb. Vsaka naslednja vrstica Q (1≤ Q≤ 1000) opisuje vsako poizvedbo. Vsaka poizvedba ima niz F ali E ali I ali S in dva cela števila, a in b, ki označujeta indekse. Pomen poizvedovalnega niza je sledi:

- F a b, pomeni, mutirajte gusarje od indeksa a do b v X ekipi.
- E a b, pomeni, mutirajte gusarje od indeksa od a do b v Y ekipe.
- I a b, pomeni, mutiram gusarje od indeksa a do b v inverznih piratov.
- S a b, Koliko gusarje ekipe X je od indeksa a do b?

(a ≤ b, 0≤ a < n, 0 ≤ b < n)

Za vsak preskus natisnite številko primera, kot kaže na primer vzorca. Potem za vsako poizvedbo, izstavite številko poizvedbe, dvopičje (:) in presledek in odgovor na poizvedbo, kot predlaga vzorec.

Primer 1: $N = 18$

101010101010001000

Primer 2: $N = 9$

111000000

Vhod:

2

2

5

10

2

1000

5

F 0 17

I 0 5

S 1 10

E 4 9

S 2 10

3

3

1

4

0

2

0

I 0 2

S 0 8

Izhod:

Case1:

Q1: 5

Q2: 1

Case 2:

Q1: 0