Andmestruktuurid

Oliver-Matis Lill

January 14, 2017

Sessiooni struktuur

- Teemad on jaotatud 2-tunnisteks plokkideks
- Ploki alguses ma pean väikese loengu, ülejäänud aeg on implementeerimiseks
- Rõhk on antud ülesannete lahendamisel
- Sessiooni korraldus ei ole range, tundge ennast mugavalt

Teemad

- Fenwick-i puu
- Lõikude puu (Segment Tree)
- Laisk laienemine (Lazy Propagation) lõikude puul
- Eriti edasijõudnutele: "Persistent Segment Tree"

Ülesanne nr. 1

- Meil on jada milles on kuni 10^5 elementi, näiteks:

4	3	6	7	5	2	1	9	7	8

- Jada peal on vaja teha järgmisi operatsioone:
  1. Suurenda mingi elemendi väärtust x võrra
  2. Anna mingile elemendile uus väärtus y
  3. Väljasta vahemiku $[l, r]$ elementide summa
- Kuni 10^5 operatsiooni

Fenwick-i puu

Idee

- Loome uue jada, vaatame indekseid kahendsüsteemis
- Vaatame uues jadas mingit kohta indeksiga i. Oletame et selle parempoolseima 1-biti väärtus on s
- Salvestame seal esiagse jada elementide $[i - s + 1, i]$ väärtuste summa
- Näiteks kui $i = 12_{10} = 1100_2$, siis $s = 100_2 = 4_{10}$, ja me salvestame sellel kohal $a_9 + a_{10} + a_{11} + a_{12}$
- Parempoolseima biti saab kätte tehtega $i\& - i$

Fenwick-i puu

Näide

f[i]	-	4	7	6	20	5	7	1	37	7	15
a[i]	-	4	3	6	7	5	2	1	9	7	8
i	0	1	2	3	4	5	6	7	8	9	10
i&-i	0	1	2	1	4	1	2	1	8	1	2

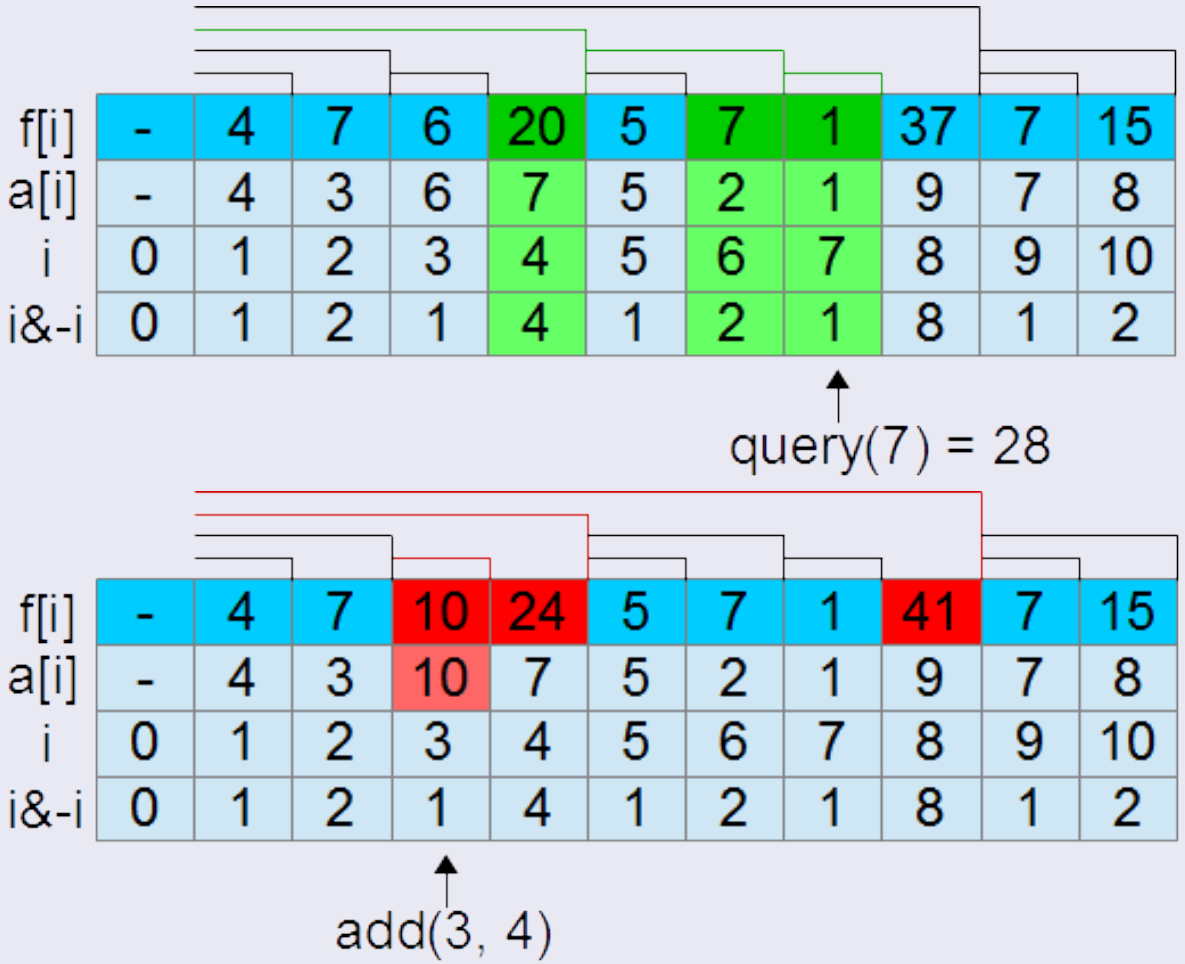
Fenwick-i puu operatsioonid

- Fenwicki puu võimaldab $O(\log n)$ ajaga väljastada vahemiku $[1, i]$ summa
- Summa arvutamiseks lisa summasse f_i väärtus ja lahuta i-st maha parempoolseim bit. Korda kuni $i = 0$
- Näiteks kui $i = 26_{10} = 11010_2$, siis vaadatavad indeksid on $11010_2, 11000_2, 10000_2$ ja summa tuleb $f_{26} + f_{24} + f_{16}$

- Lisaks saab $O(\log n)$ ajaga suurendada elemendi i väärtust x võrra
- Lisamiseks, suurenda f_i väärtust x võrra ja liida i-le parempoolseim bit. Korda kuni i on jada pikkusest suurem
- Näiteks kui $i = 45_{10} = 101101_2$, siis läbitakse indeksid $101101_2, 101110_2, 110000_2, 1000000_2$ ja uuendatakse $f_{45}, f_{46}, f_{48}, f_{64}$

Fenwick-i puu operatsioonid

Näide



Ülesanne nr. 2

- Meil on kuni 10^5 jada elementi ja operatsiooni
- Operatsioonid on järgmised
  1. Suurenda mingi elemendi väärtust x võrra
  2. Anna mingile elemendile uus väärtus y
  3. Väljasta vahemiku $[l, r]$ elementide **maksimum**
- Kas siin töötaks Fenwick-i puu?

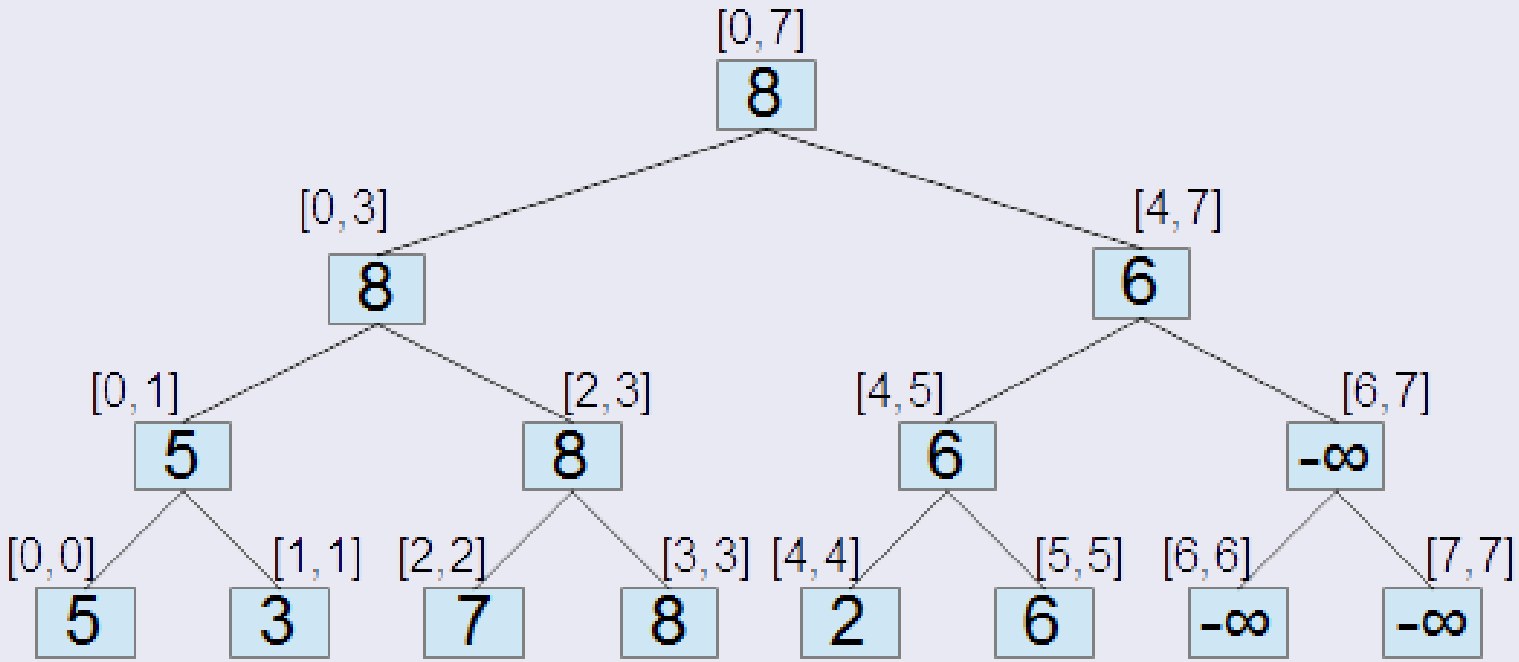
Lõikude puu

Idee

- Kasuta perfektset kahendpuud, lehtedes hoia jada elemendid
- Ülejäänud tipud hoiavad oma alamate maksimumi/summat/vms
- Tulemusena iga tipp hoiab kõikide oma alampuu lehtede maksimumi, mis vastab mingile esialgse jada lõigule

Lõikude puu

Näide

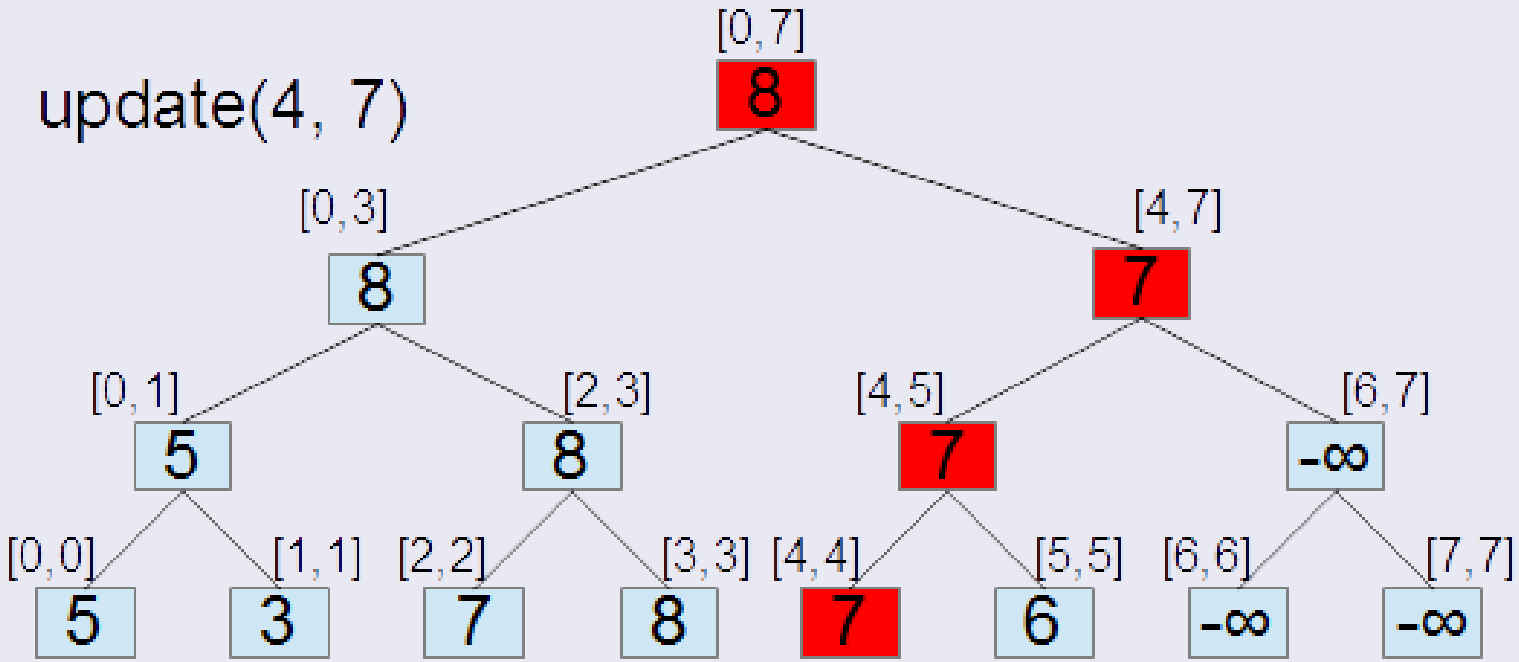


Lõikude puu operatsioonid

- Lõikude puus saab jada elemendi väärtust uuendada järgmise rekursiivse algoritmiga, mis käivitatakse juurest:
  1. Kui tipp on leht, siis uuenda väärtus ja lõpeta, muidu jätka
  2. Rekursiivselt käivita algoritm selle alama peal, mis hoiab uuendatavat elementi
  3. Uuenda praeguse tipu väärtust alamate uute väärtuste põhjal
- Algoritm läbib tee juurest uuendatava leheni ja uuendab kõikidel teel olevate tippude väärtused
- Kuna tegemist on perfektse kahendpuuga, siis tee pikkus ja algoritmi keerukus on O(log n)

Lõikude puu operatsioonid

Näide

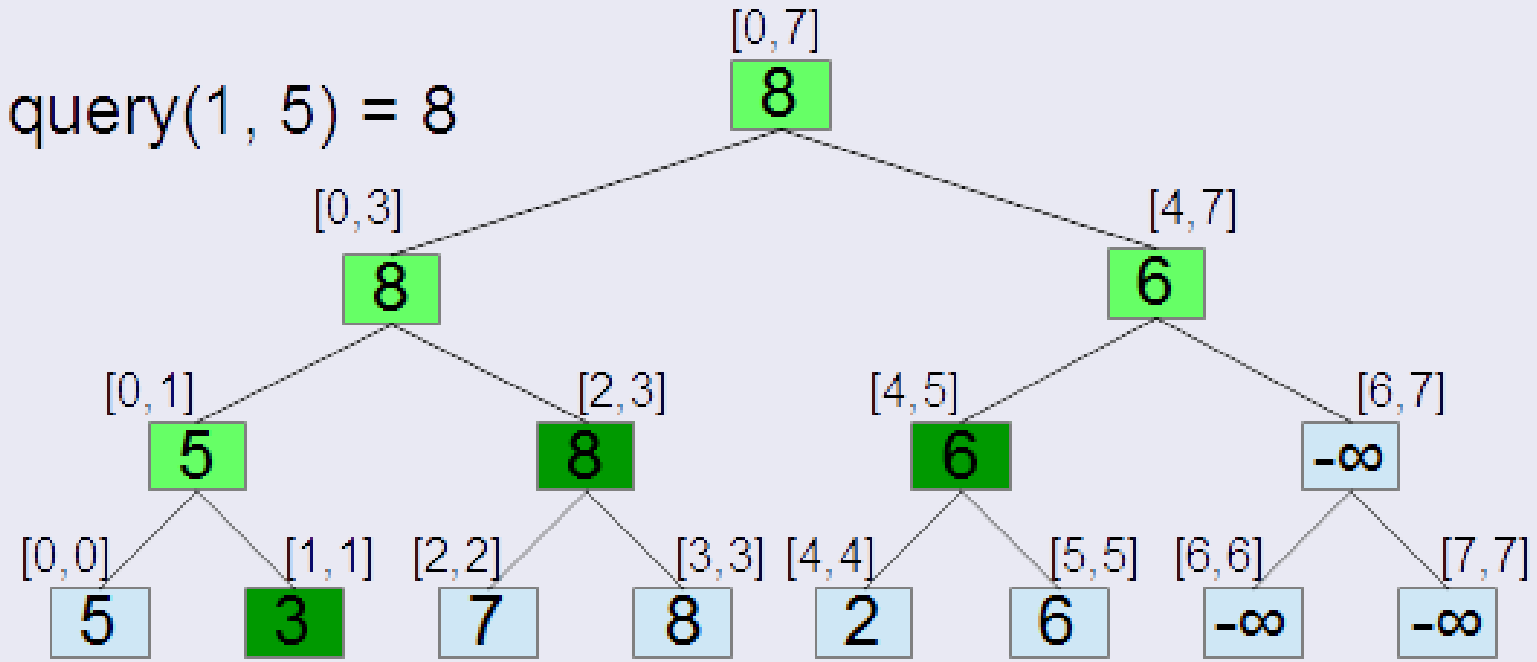


Lõikude puu operatsioonid

- Vahemiku elementide maksimumi saab leida järgmise rekursiivse algoritmiga, mis käivitatakse samuti juurest:
  1. Kui tipu vahemik on täielikult otsitava vahemiku sees, siis väljasta tipu väärtus, muidu jätka
  2. Kui vasak alam omab ühisosa otsitava vahemikuga, siis rekursiivselt käivita algoritm selle alama peal
  3. Kui parem alam omab ühisosa vahemikuga, siis käivita algoritm selle peal
  4. Väljasta alamatelt väljastatud väärtuste maksimum
- Selle algoritmi keerukus on samuti O(log n)
- Peale esimest hargnemist on kõikidel järgnevatel hargnemistel kas üks alam täielikult vahemiku sees, või teine täielikult vahemikust väljas
- Tulemusena on igal astmel ülimalt kaks tippu mida algoritm läbib ja kus see ei väljasta kohe tipu väärtuse

Lõikude puu operatsioonid

Näide



Ülesanne nr. 3

- Siin ka kuni 10^5 jada elementi ja operatsiooni
- Operatsioonid on järgmised:
  1. Suurenda vahemikus $[l, r]$ olevate elementide väärtused d võrra
  2. Anna vahemiku $[l, r]$ igale elemendile väärtuseks x
  3. Väljasta vahemikus $[l, r]$ olevate elementide summa
- Kuidas vahemike manipuleerimist siin lubada?

Laisk laienemine

Idee

- Kui operatsioon mõjutab tervenisti mingi tipu vahemikku, siis on kerge selle väärtust uuendada
- Uuendame rekursiivselt ära kõik tipud, mis on täielikult opereeritava vahemiku sees, ärme nende alamaid veel muuda
- Märgistame uuendatud tippude alamatele et neid on vaja uuendada. Uuendame alamad alles hiljem kui nende väärtust vaja on
- Seega on vaja kahte funktsiooni:
  1. Märgistamisfunktsioon: märgistab igal operatsioonil, mis tipud tuleb uuendada
  2. Laienemisfunktsioon: rakendab tipule märgistatud operatsioonid ja saadab need alamatele edasi

Laisk laienemine

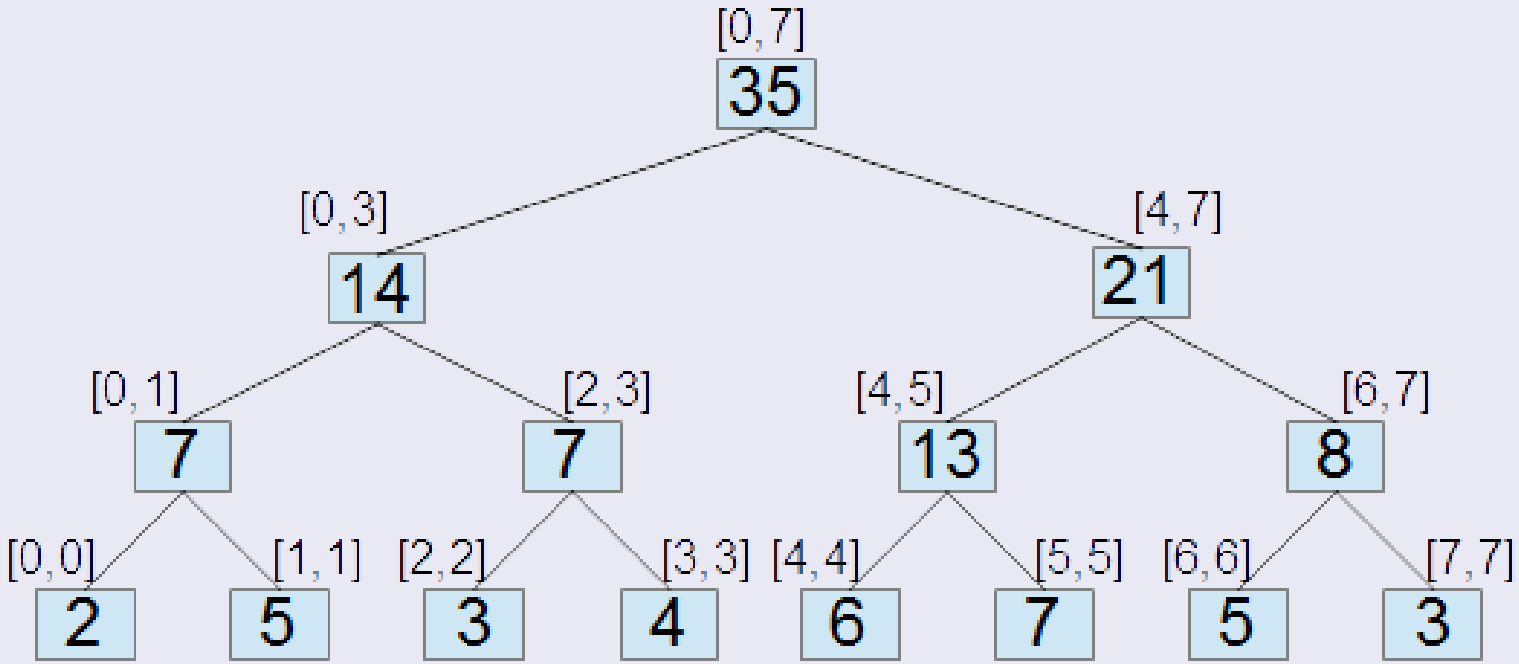
- Märgistamisfunktsioon käivitatakse juurest ja see märgistab vastavatele tippudele mis operatsiooni neil rakendada tuleb.
- Funktsiooni algoritm on järgmine:
  1. Kui tipp on täielikult opereeritava vahemiku sees, siis uuenda tipp, märgista alamatele operatsioon ja lõpeta, muidu jätka
  2. Kui vasak alam omab ühisosa opereeritava vahemikuga, siis rekursiivselt käivita funktsioon selle peal
  3. Kui parem alam omab ühisosa, siis käivita selle peal antud funktsioon
  4. Uuenda tipu väärtus alamate uute väärtuste põhjal
- Algoritm läbib puu sama moodi nägu varemkirjeldatud päringufunktsioon, ainsa vahega et tulemuse väljastamise asemel märgistatakse operatsioone

Laisk laienemine

- Laienemisfunktsioon käivitatakse tipul alati kui seda läbitakse või selle väärtust küsitakse.
- Funktsiooni algoritm on järgmine:
  1. Kui tipul on märgistatud väärtustamisoperatsioon, siis tühista alamatelt kõik operatsioonid ja märgista need selle väärtustamisega
  2. Rakenda väärtustamisoperatsioon tipule ja tühista operatsiooni märgistus
  3. Kui tipul on märgistatud suurendamisoperatsioon, siis saada see suurendamisoperatsioon alamatele edasi, varasemad operatsioonid jäävad neil alles
  4. Rakenda suurendamisoperatsioon tipule ja tühista selle märgistus
- Keerukus on ilmselt $O(1)$
- See funktsioon tagab et tipud oleks õigete väärtuste peale uuendatud ainult siis kui meid nende väärtus huvitab

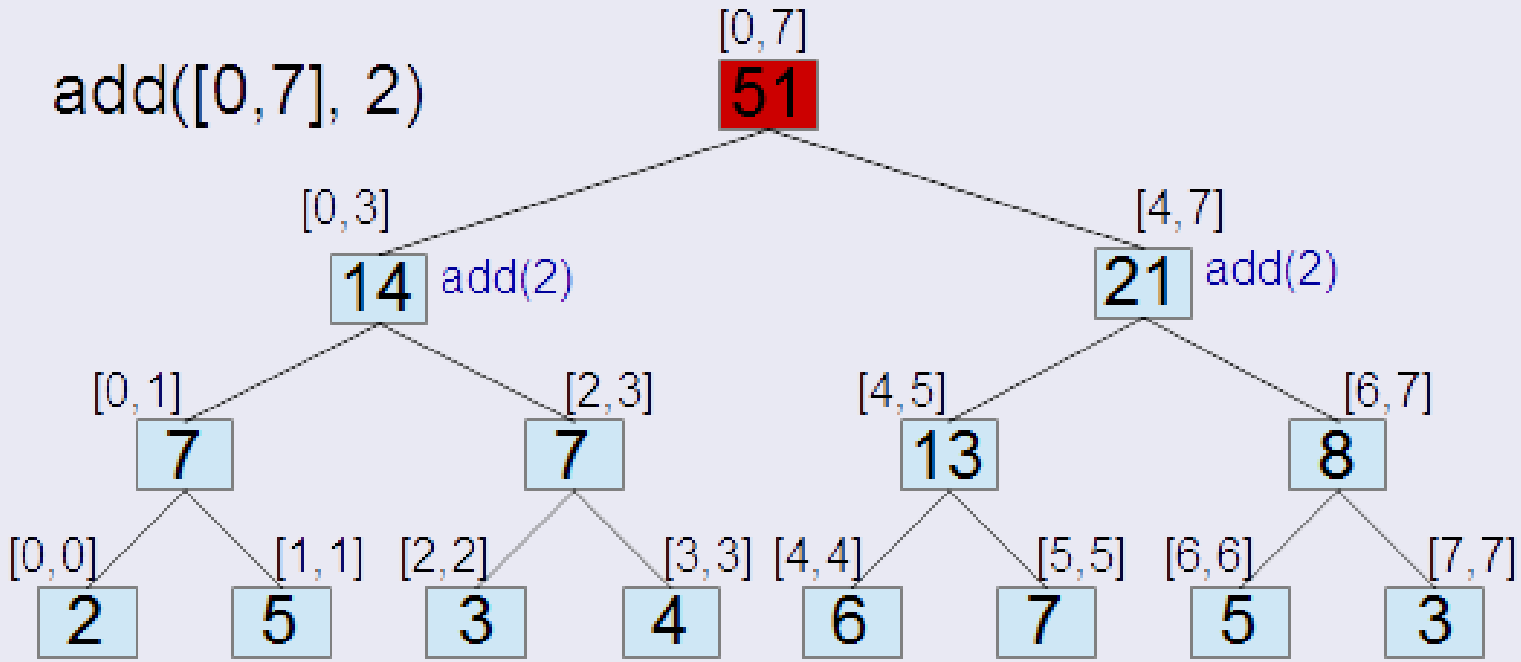
Laisk laienemine

Näide



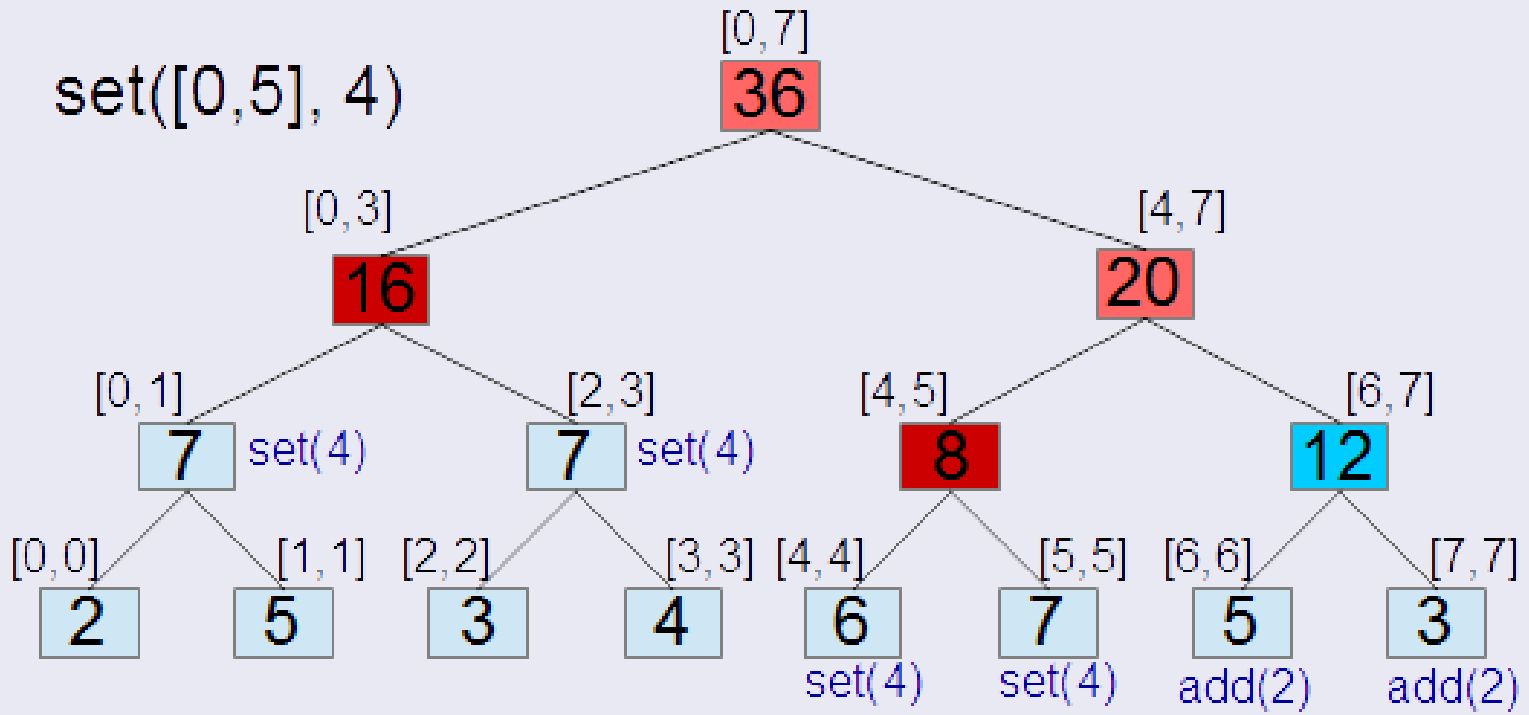
Laisk laienemine

Näide



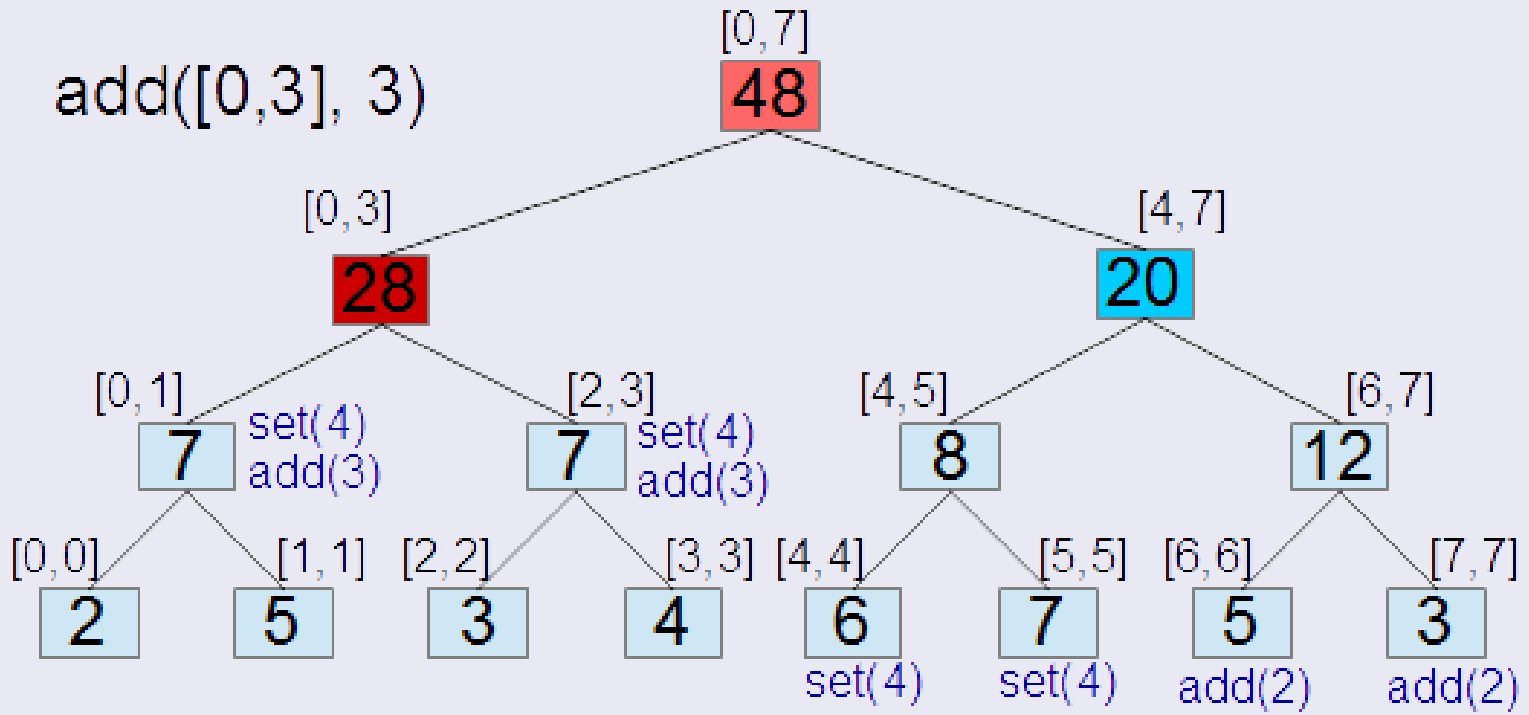
Laisk laienemine

Näide



Laisk laienemine

Näide



Laisk laienemine

Näide