Estruturas Persistentes

Lucas Turci

GEMA - ICMC

O que é persistência?

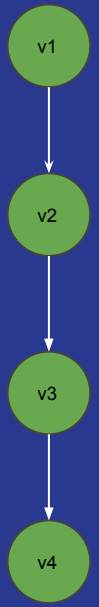
Cada modificação em uma estrutura persistente cria uma versão nova. As versões anteriores *persistem* na memória.

O que é persistência?

Seja T um TAD que implemente uma operação de consulta `query` e uma de modificação `update`.

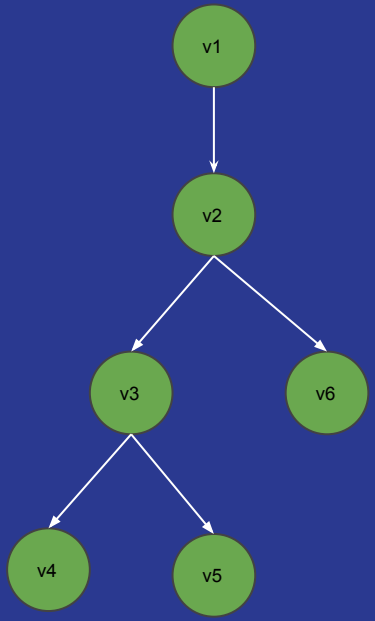
- `T` começa na versão `0`
- `T.update(v?)` retorna uma nova versão
- `T.query(v)` retorna a consulta da versão `v`

Persistência Parcial



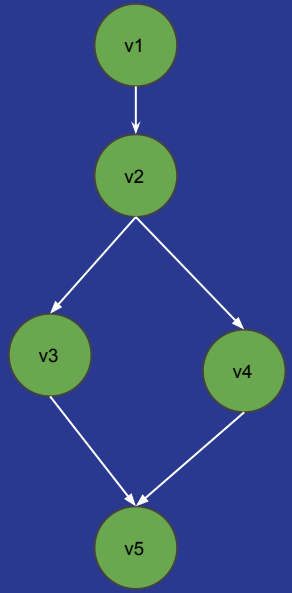
- Estrutura linear
- Versões antigas são Read-Only
- Só a versão atual pode ser atualizada
- Rollback

Persistência Total



- Versões antigas podem ser lidas e modificadas
- Estrutura de árvore
- Ramificações (“linhas do tempo”)

Persistência Confluente



- Nova feature: ***merge***
- Estrutura em DAG
- É o tipo do git
- Meio subjetivo o que é considerado *merge*

Ainda existe o tipo: “persistência funcional”, que não será discutido aqui

Aggie.io

Persistência Parcial

Ex.: Array

Aggie.io

Persistência Total

Ex.: Pilha

Como fazer persistência total em array?

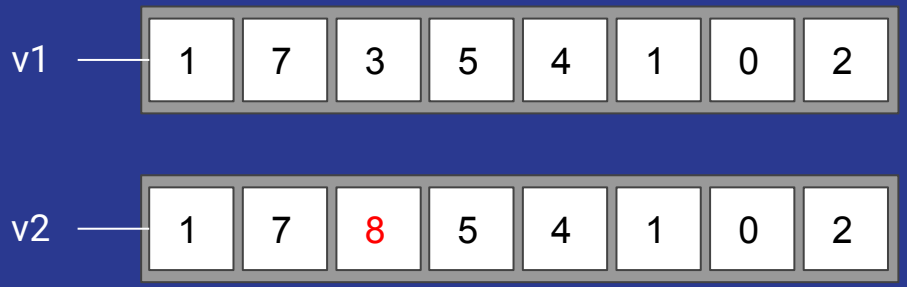
Voltando para a solução naive

Como fazer persistência total em array?

Voltando para a solução naive

Como fazer persistência total em array?

Voltando para a solução naive



O(N) fields precisam ser copiados :(

e

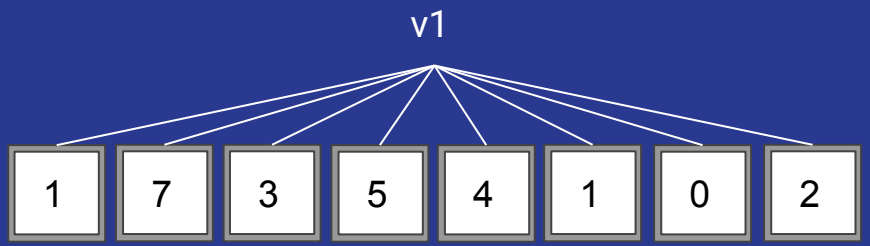
O(1) ponteiro é criado

Como fazer persistência total em array?

Outra ideia:

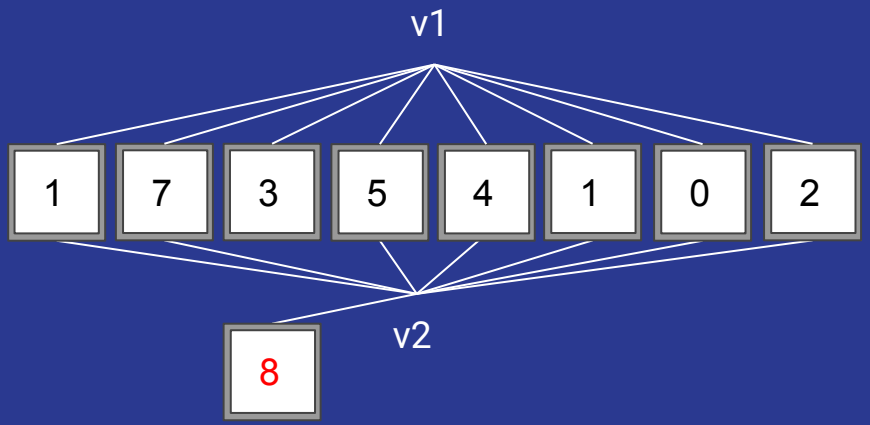
Como fazer persistência total em array?

Outra ideia:



Como fazer persistência total em array?

Outra ideia:



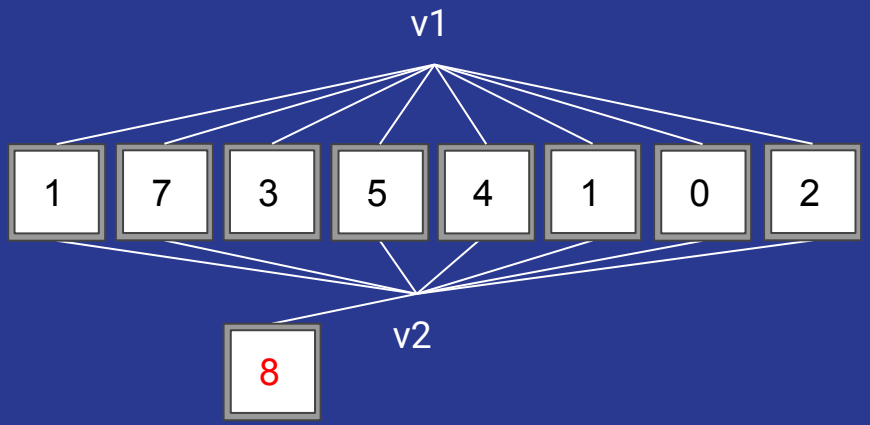
O(1) field é criado

e

O(N) ponteiros são criados :(

Como fazer persistência total em array?

Outra ideia:



O(1) field é criado

e

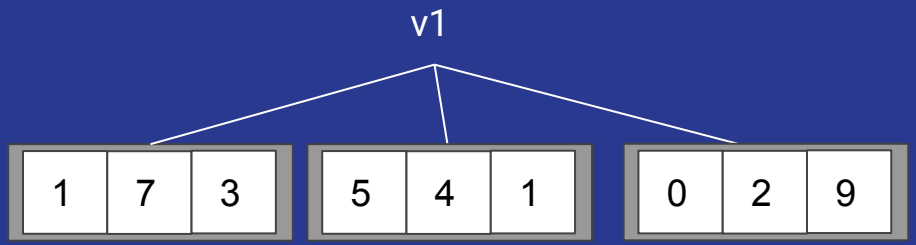
O(N) ponteiros são criados :(

Como fazer persistência total em array?

~ Equilíbrio ~

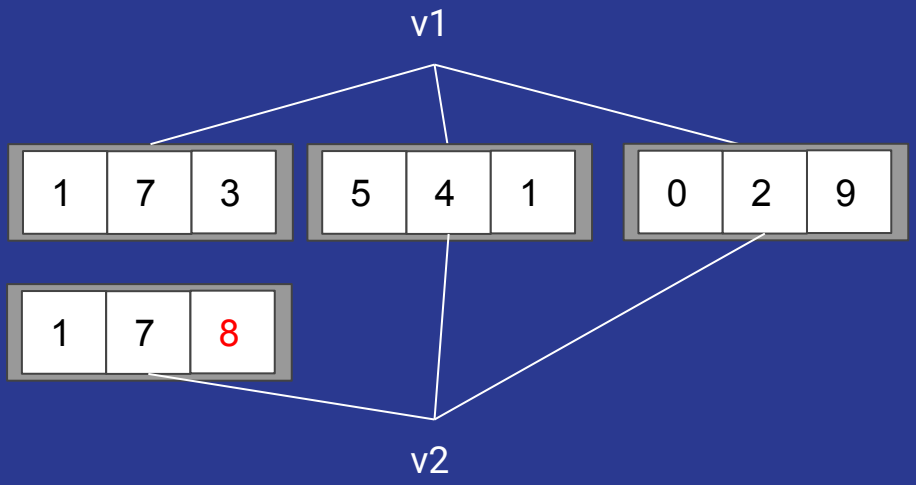
Como fazer persistência total em array?

~ Equilíbrio ~



Como fazer persistência total em array?

~ Equilíbrio ~



$O(N^{\frac{1}{2}})$ fields são copiados

e

$O(N^{\frac{1}{2}})$ ponteiros são criados


BALANCE IS KEY.

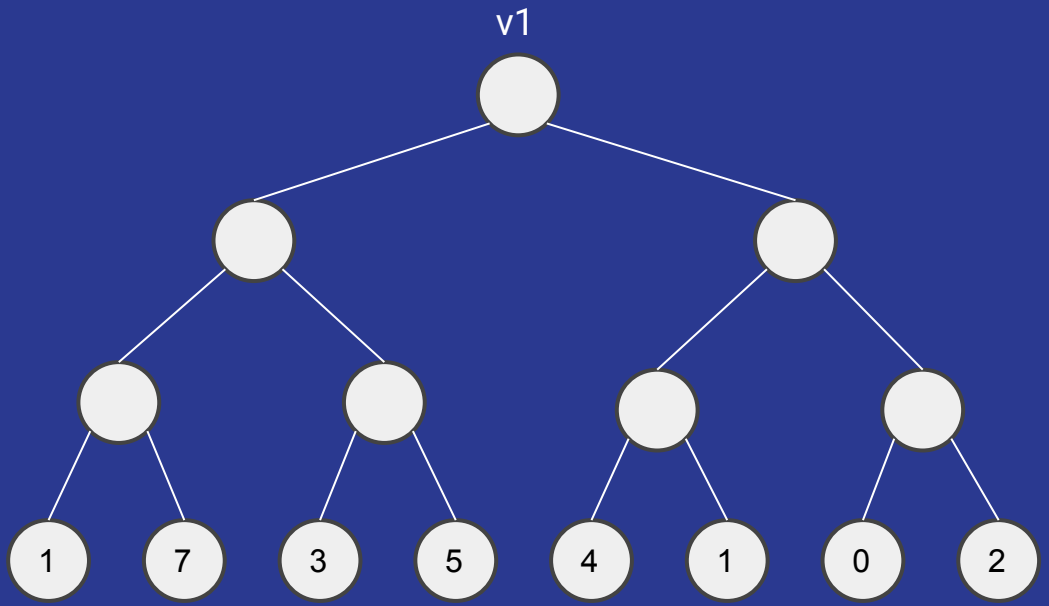
Como fazer persistência total em array?

Segtreeeee

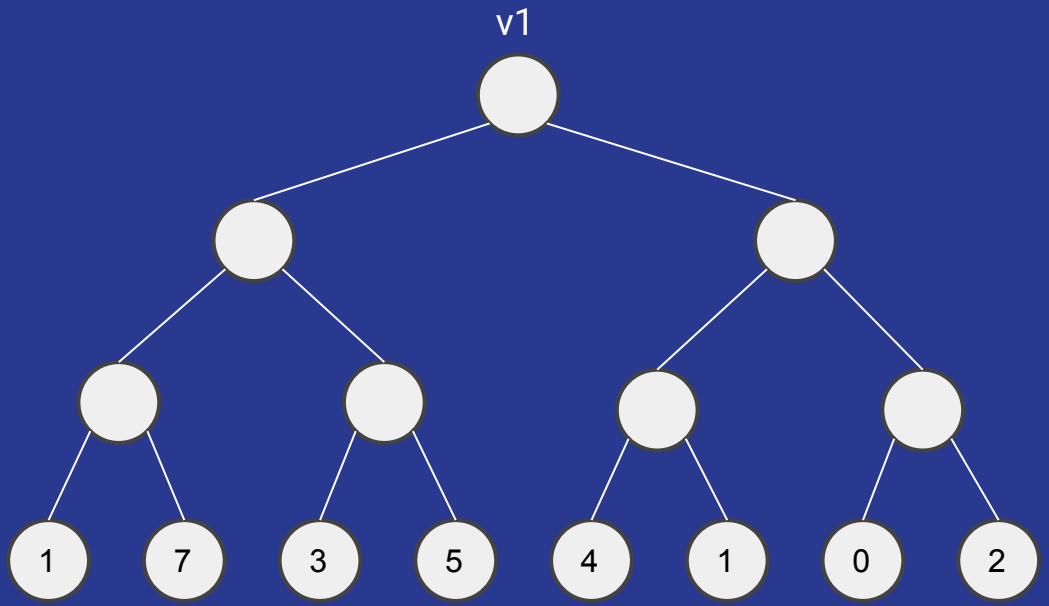
Como fazer persistência total em array?

Segtreeeee

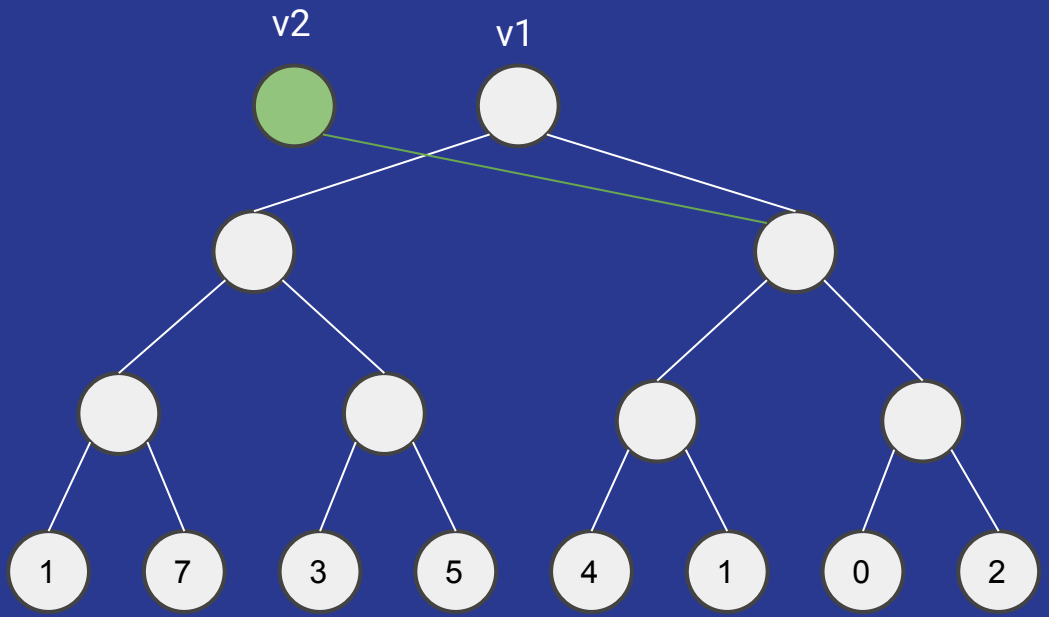
- Cada nó da segtree representa a junção de dois subarrays
- Um update gera O(lgN) novos fields (nós) e O(lgN) novos ponteiros


v1
1
7
3
5
4
1
0
2

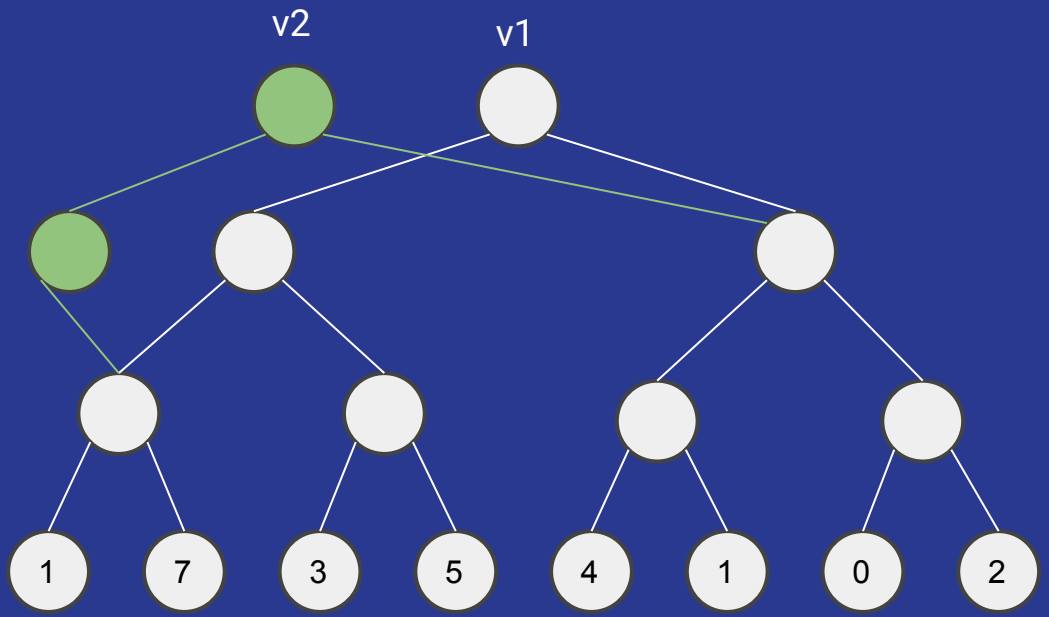
update(2, 8)



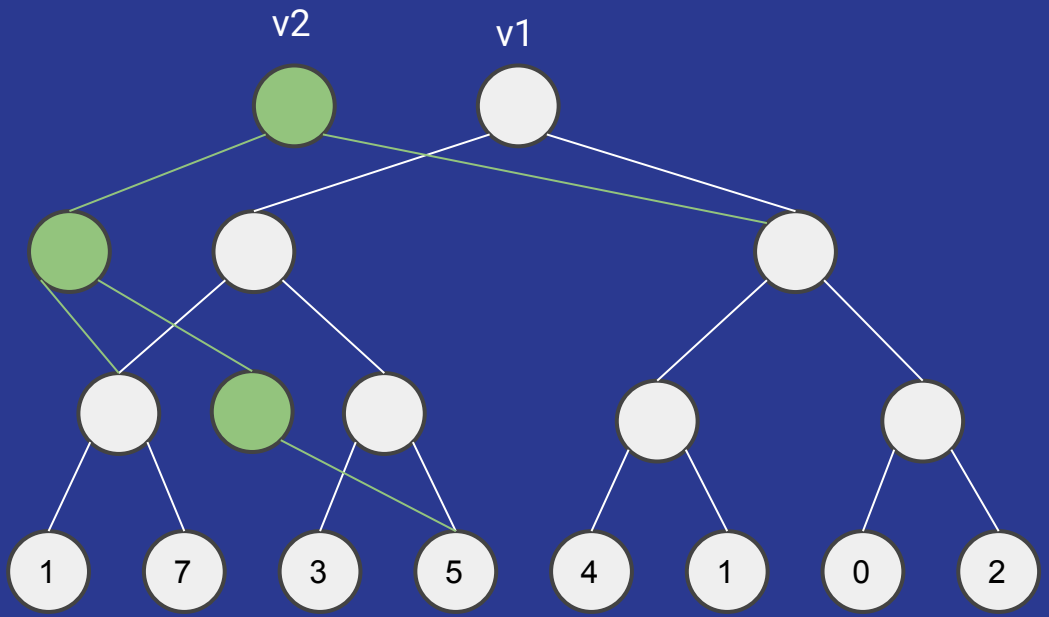
update(2, 8)



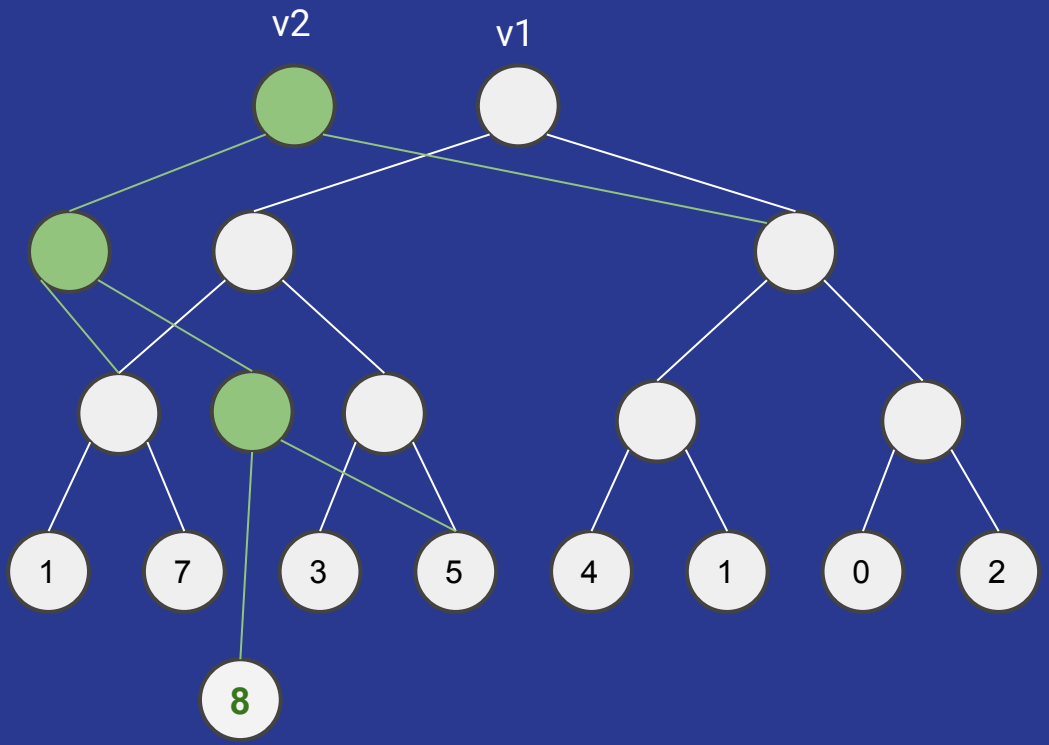
update(2, 8)



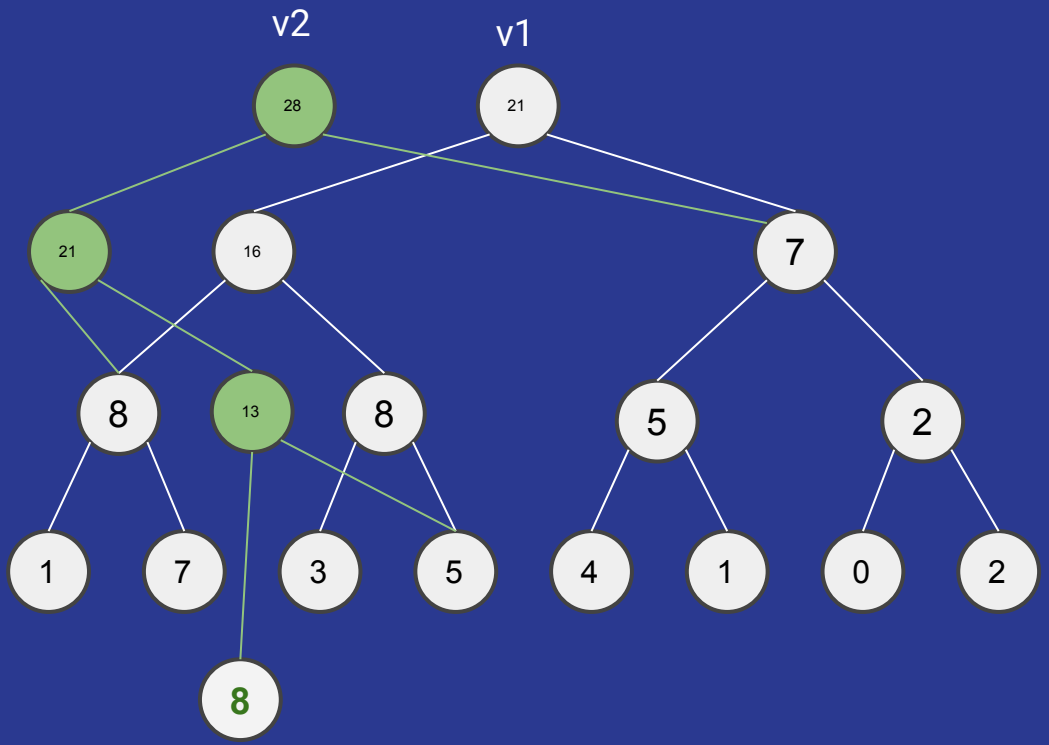
update(2, 8)



update(2, 8)



update(2, 8)



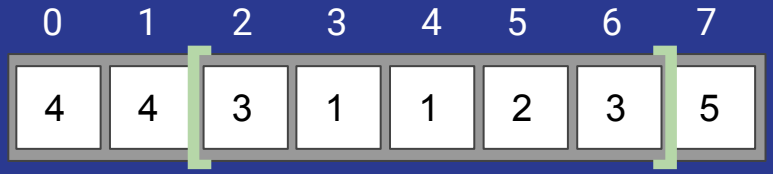
Implementação

https://cses.fi/problemset/task/1737

Problema

Dado um array de inteiros de tamanho `N` e `Q` queries, do tipo `<L, R>`, responda: quantos elementos distintos existem no subarray `A[L...R]`?

Query: [2, 6]



Query: [2, 6]

Só considera a última aparição de cada elemento

Query: [2, 6]

Só considera a última aparição de cada elemento
E se A[7] fosse 3?

Query: [2, 6]

Só considera a última aparição de cada elemento até R

Query: [2, 6]

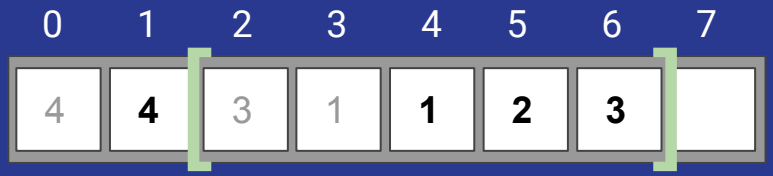
Só considera a última aparição de cada elemento até R
Como contar?

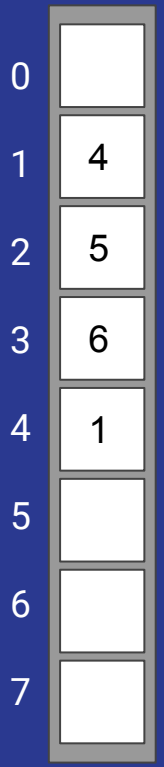
Query: [2, 6]

Só considera a última aparição de cada elemento até R
Como contar? Mantém uma tabela.

0	
1	4
2	5
3	6
4	1
5	
6	
7	

Query: [2, 6]

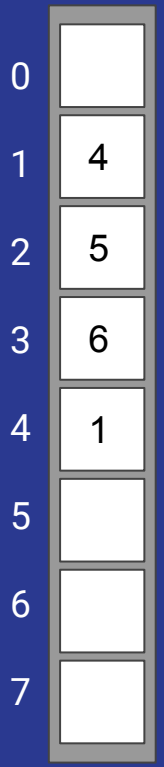




Só considera a última aparição de cada elemento até R
Como contar? Mantém uma tabela.

O problema vira: **quantos números nessa tabela são maiores que L?**

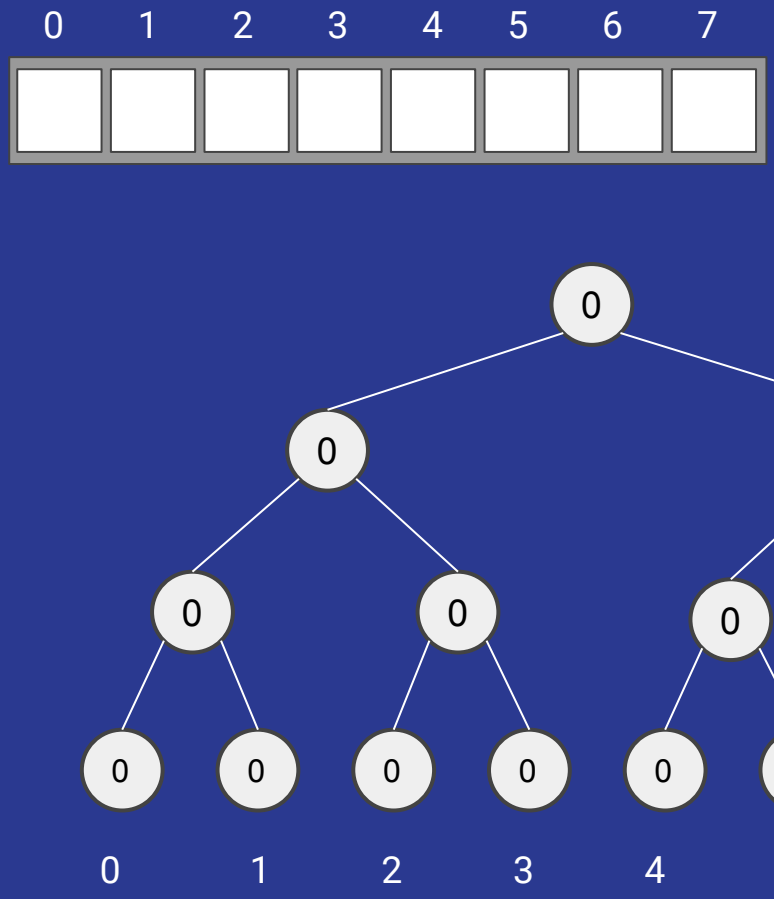
Query: [2, 6]

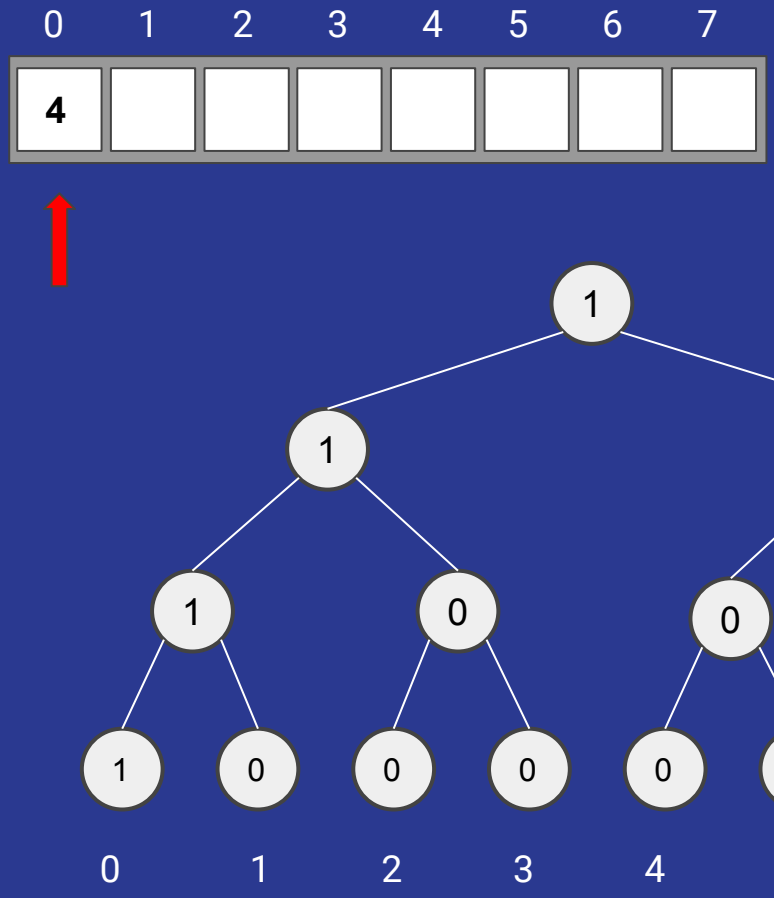


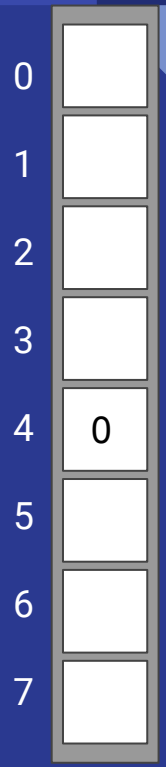
Só considera a última aparição de cada elemento até R
Como contar? Mantém uma tabela.

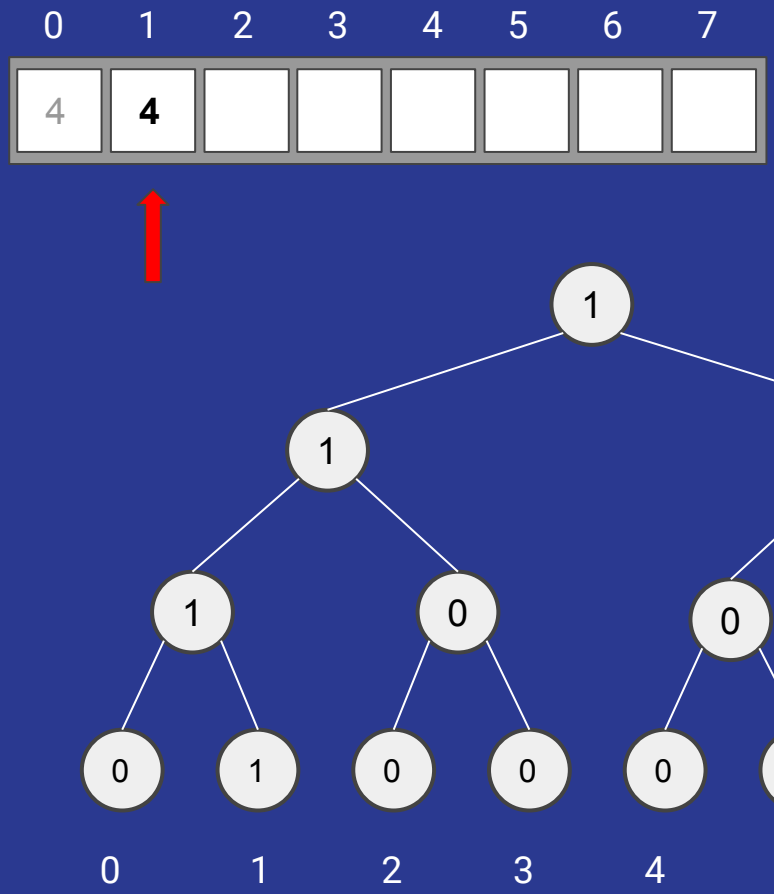
O problema vira: **quantos números nessa tabela são maiores que L?**

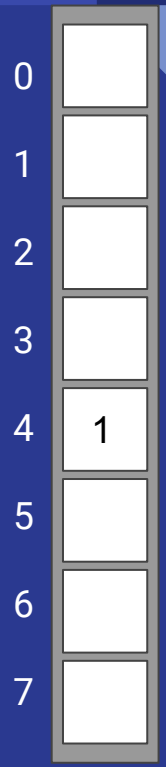
Solução: mantém uma segtree

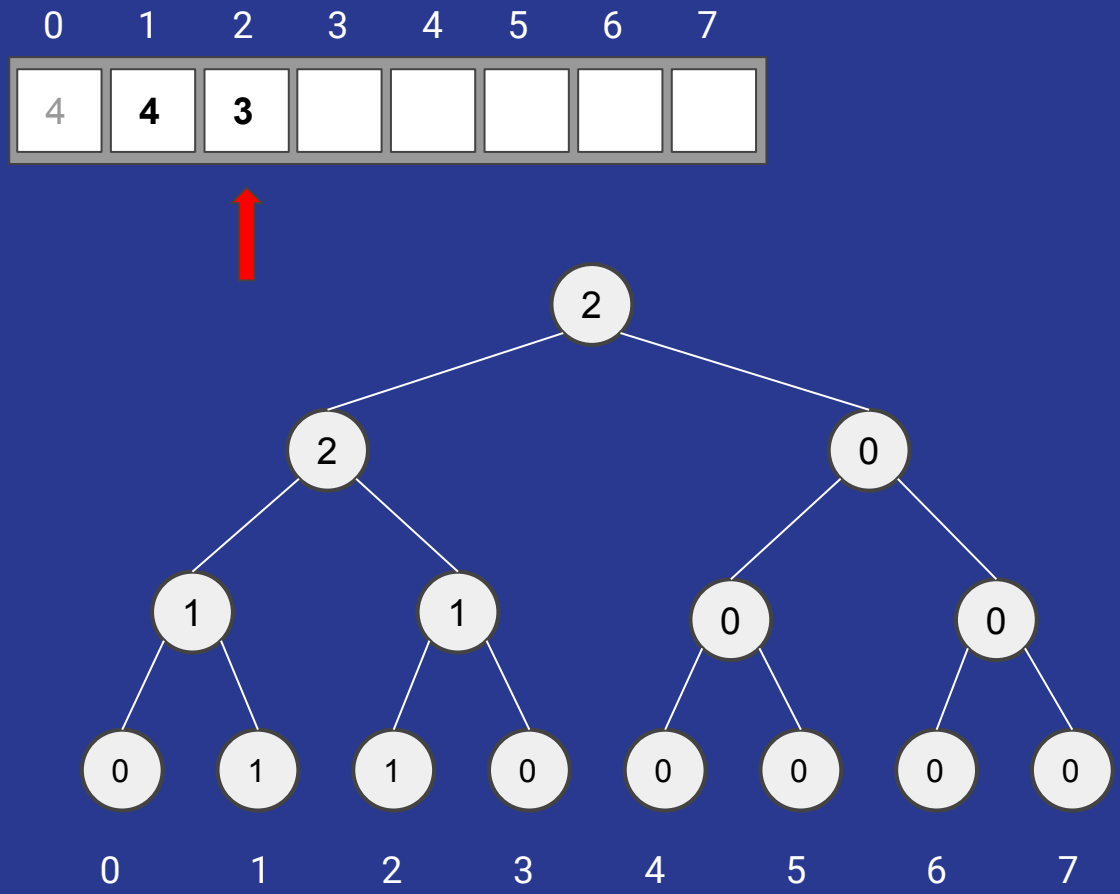

0 1 2 3 4 5 6 7
0
0 0
0 0 0 0
0 0 0 0 0 0 0 0
0 1 2 3 4 5 6 7
0
1
2
3
4
5
6
7

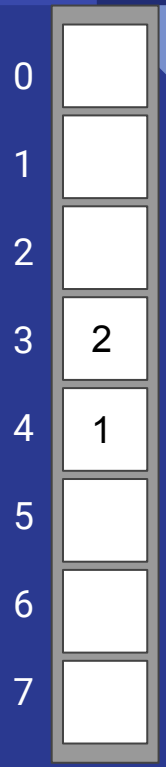

0 1 2 3 4 5 6 7
4
1
1 0
1 0 0 0
1 0 0 0 0 0 0 0
0 1 2 3 4 5 6 7

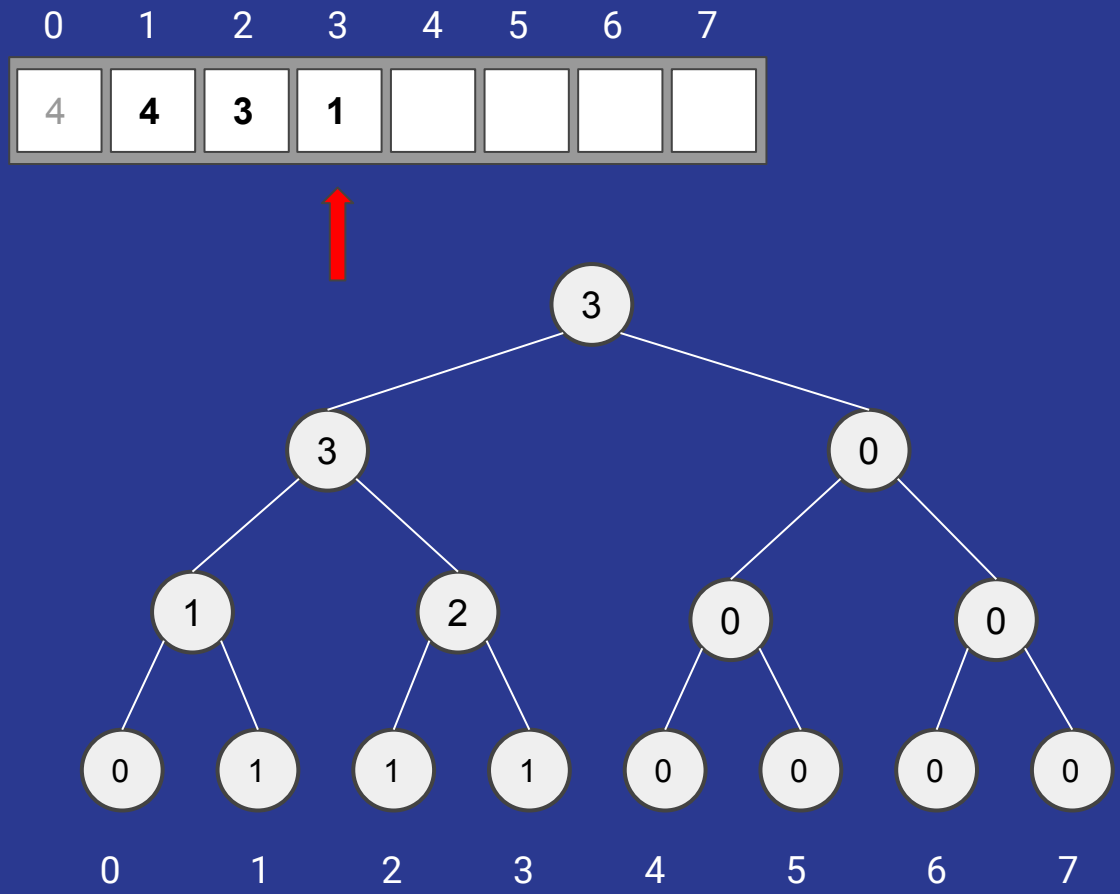

0
1
2
3
4
0
5
6
7

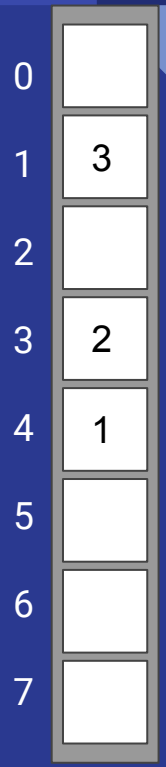

0 1 2 3 4 5 6 7
4 4
1
1 0
1 0 0 0
0 1 0 0 0 0 0 0
0 1 2 3 4 5 6 7

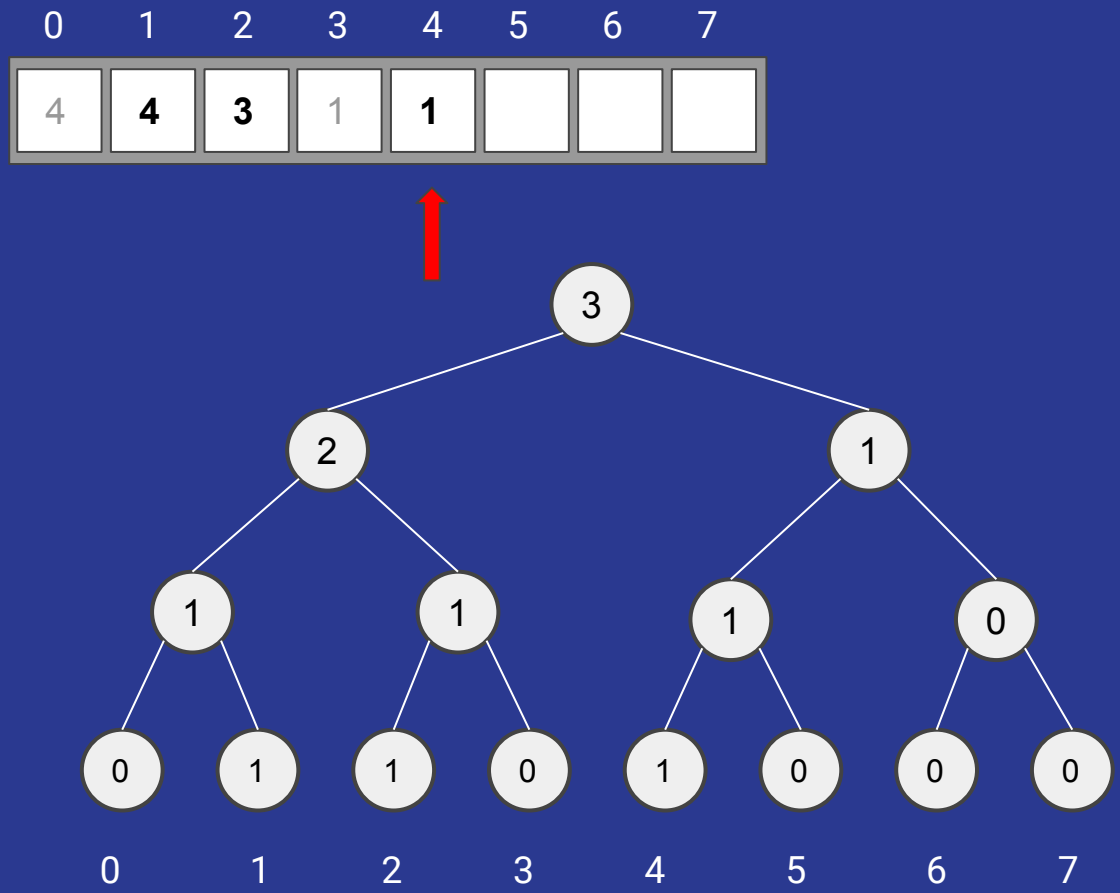

0
1
2
3
4 1
5
6
7

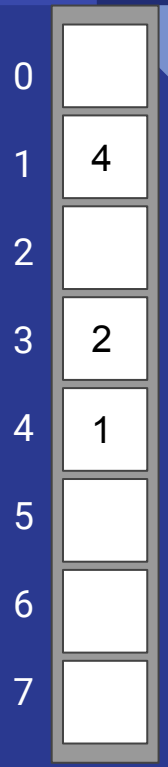

0 1 2 3 4 5 6 7
4 4 3
2
2 0
1 1 0 0
0 1 1 0 0 0 0 0
0 1 2 3 4 5 6 7

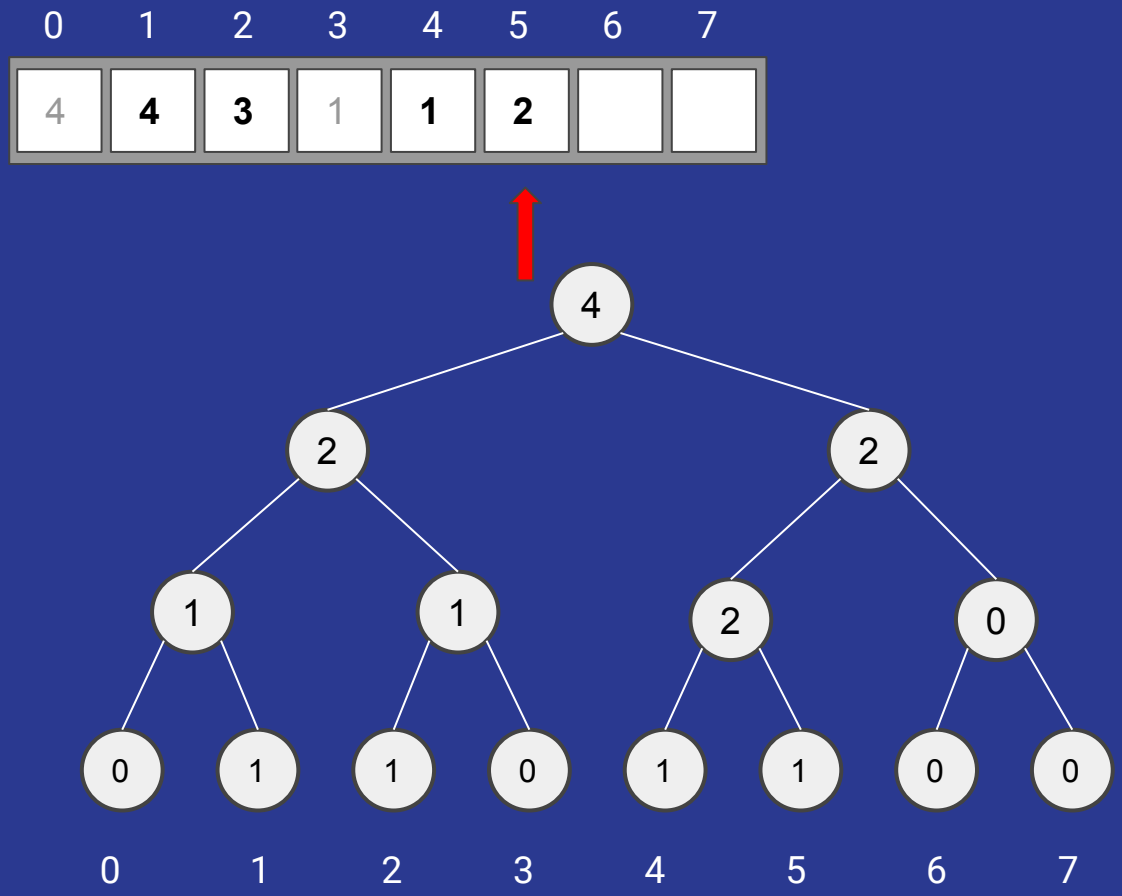

0
1
2
3 2
4 1
5
6
7

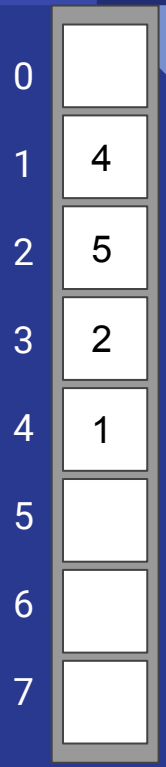

0 1 2 3 4 5 6 7
4 4 3 1
3
3 0
1 2 0 0
0 1 1 1 0 0 0 0
0 1 2 3 4 5 6 7


0
1 3
2
3 2
4 1
5
6
7

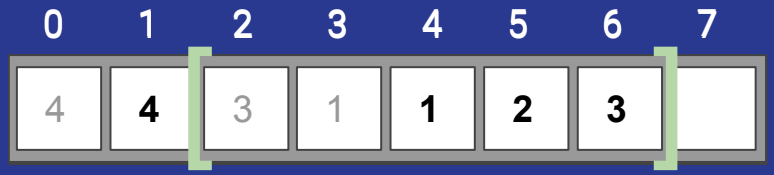

0 1 2 3 4 5 6 7
4 4 3 1 1
3
2 1
1 1 1 0
0 1 1 0 1 0 0 0
0 1 2 3 4 5 6 7

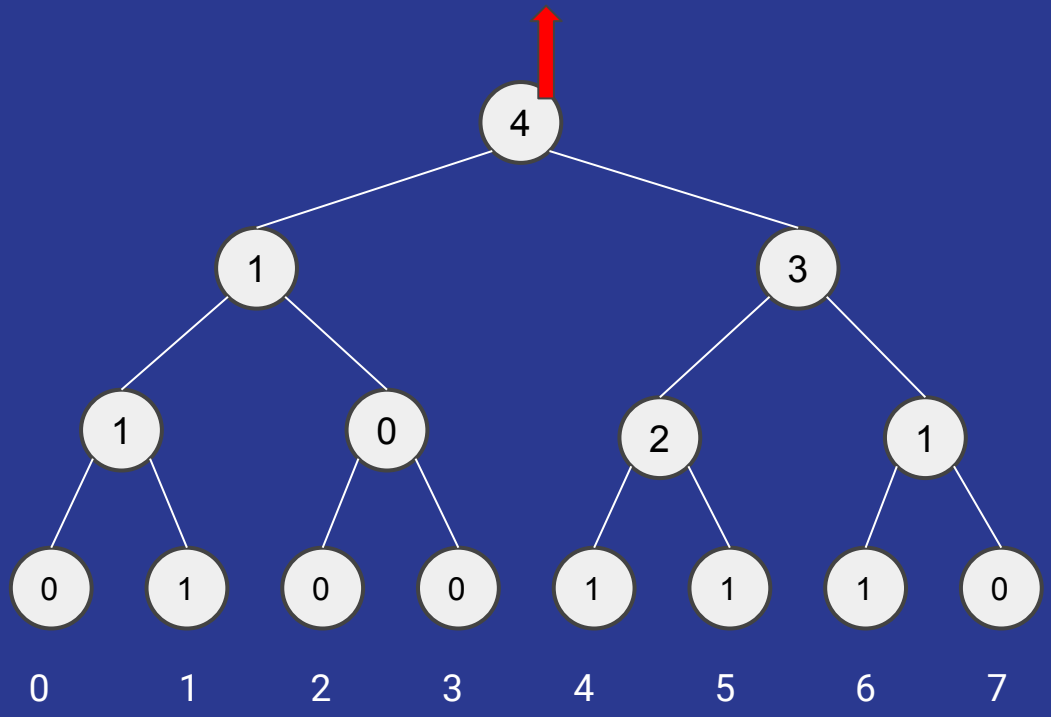

0
1 4
2
3 2
4 1
5
6
7

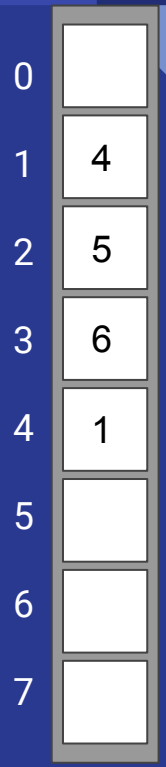

0 1 2 3 4 5 6 7
4 4 3 1 1 2
4
2 2
1 1 2 0
0 1 1 0 1 1 0 0
0 1 2 3 4 5 6 7


0
1 4
2 5
3 2
4 1
5
6
7

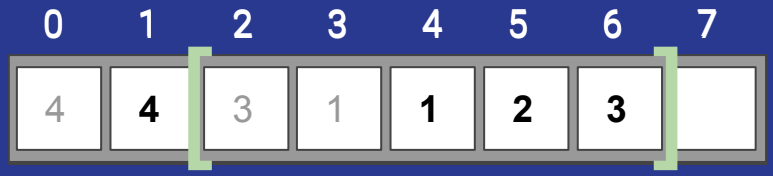
Query: [2, 6]

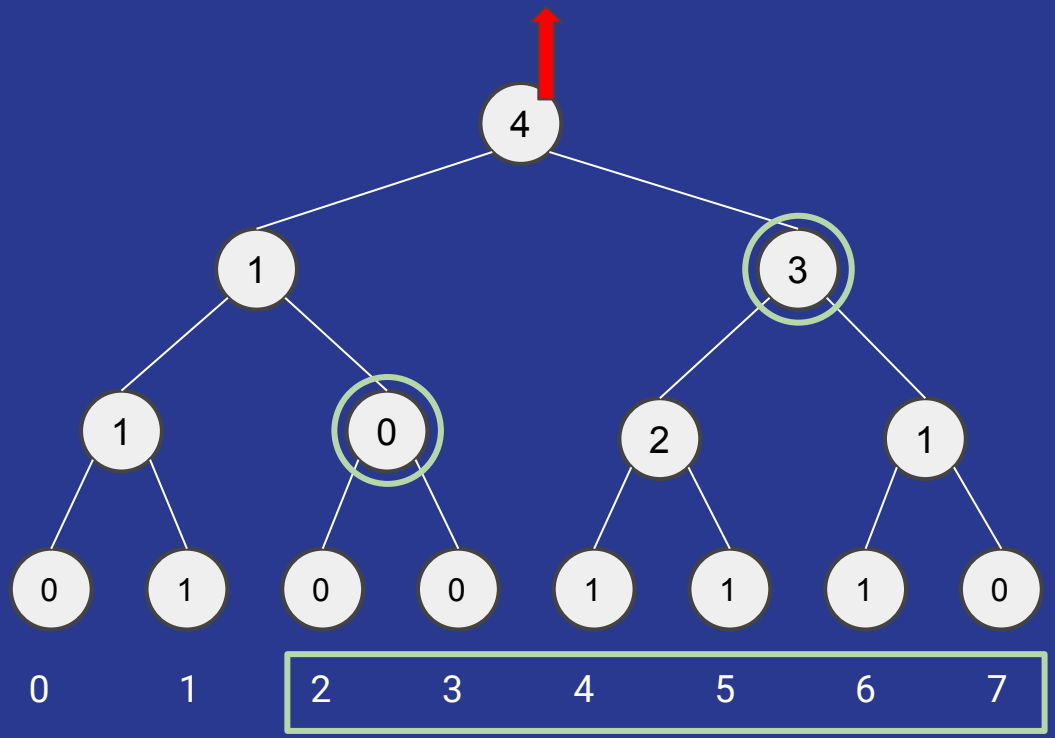

0 1 2 3 4 5 6 7
4 4 3 1 1 2 3

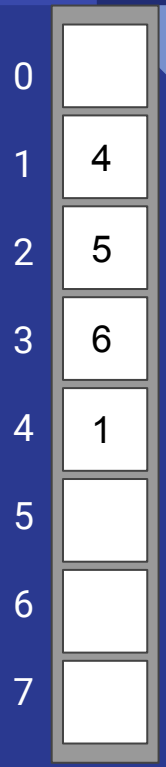

4
1 3
1 0 2 1
0 1 0 0 1 1 1 0
0 1 2 3 4 5 6 7


0
1 4
2 5
3 6
4 1
5
6
7

Query: [2, 6]


0 1 2 3 4 5 6 7
4 4 3 1 1 2 3


4
1 3
1 0 2 1
0 1 0 0 1 1 1 0
0 1 2 3 4 5 6 7


0
1 4
2 5
3 6
4 1
5
6
7

Problema

E se as queries forem online?

Problema

E se as queries forem online?

Segtree Persistente

Problema

E se as queries forem online?

Segtree Persistente

persiste as segtrees resultantes de cada iteração.

Para resolver a query <L, R>, usa a versão R

Como fazer persistência total em ~~array~~ vector?

Troca segtree por **Treap** (aula futura)

Como fazer persistência ~~total~~ parcial em ~~array~~ sets?

Segtree resolve!

Implicit Sets

Queries:

- `s.in(x)`
- `s.kth(k)`
- `s.lower/upper_bound(x)`

Updates:

- `s.insert(x)`
- `split(s1, s2, k) // s1 fica com <= k e s2 com > k`
- `split(s1, s2, k) // s1 fica com primeiros k e s2 com resto`
- `merge(s1, s2) // retorna s1 U s2`

Implicit Sets

Queries:

- `s.in(x)`
- `s.kth(k)`
- `s.lower/upper_bound(x)`

Pra resolver esses, segtree de soma é suficiente

Implicit Sets

Queries:

- `s.in(x)`
- `s.kth(k)`
- `s.lower/upper_bound(x)`

Pra resolver esses, segtree de soma é suficiente
split e merge???

Implicit Sets

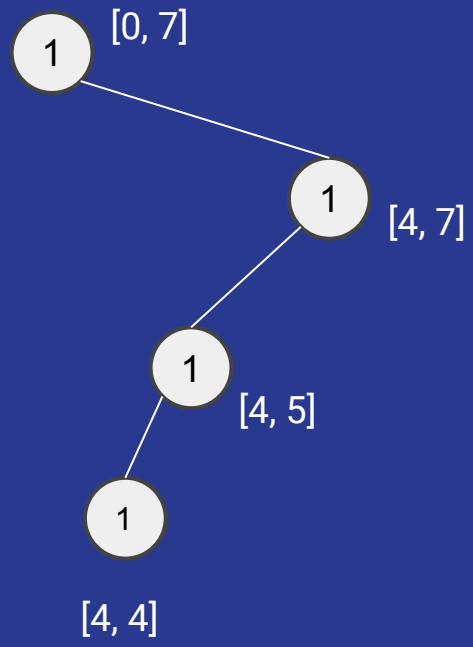
Queries:

- `s.in(x)`
- `s.kth(k)`
- `s.lower/upper_bound(x)`

Pra resolver esses, segtree de soma é suficiente
split e merge???

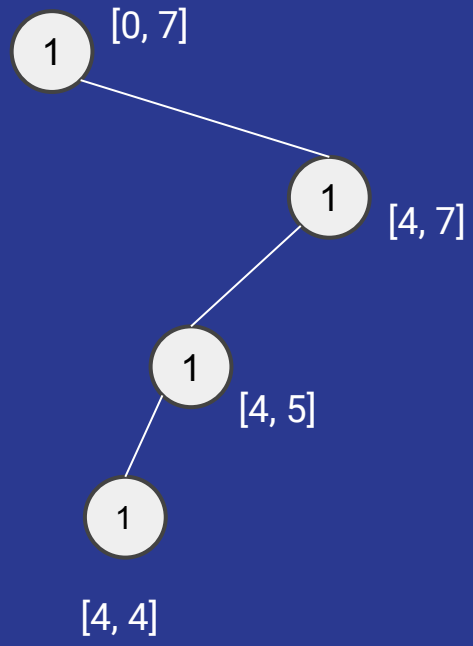
Cria nós *on demand* (segtree esparsa)

Implicit Sets



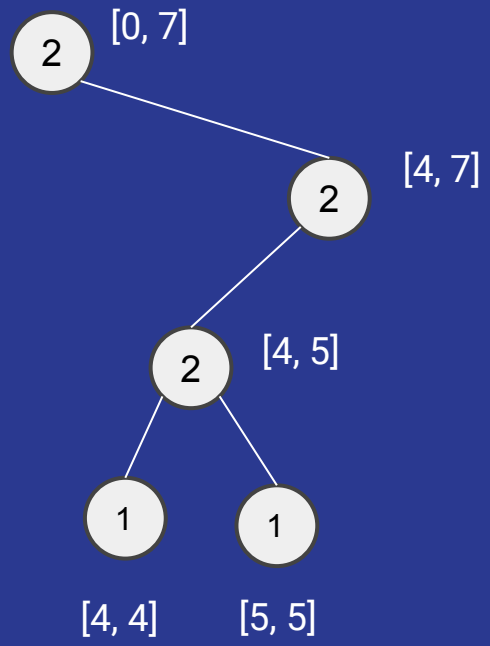
Implicit Sets

Insert(5)



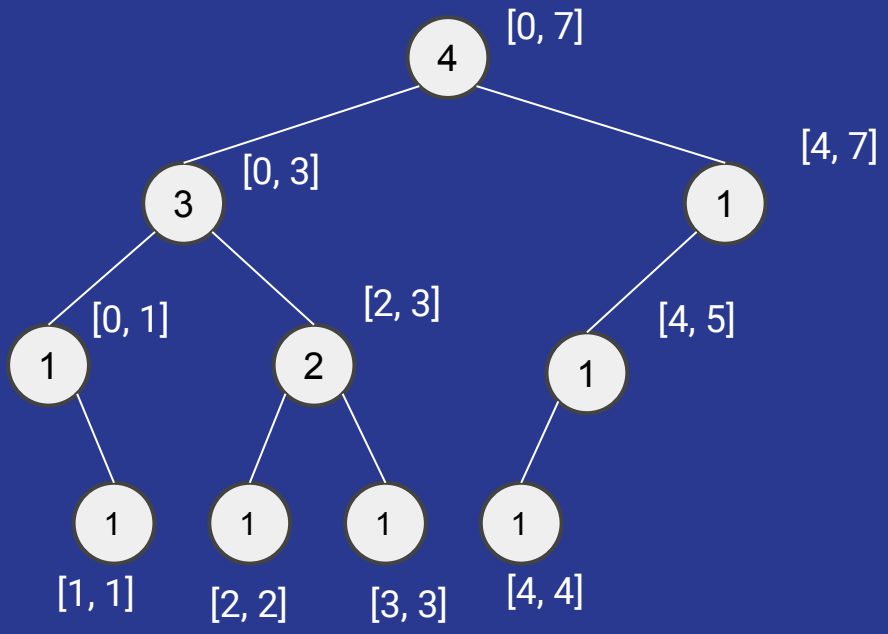
Implicit Sets

Insert(5)



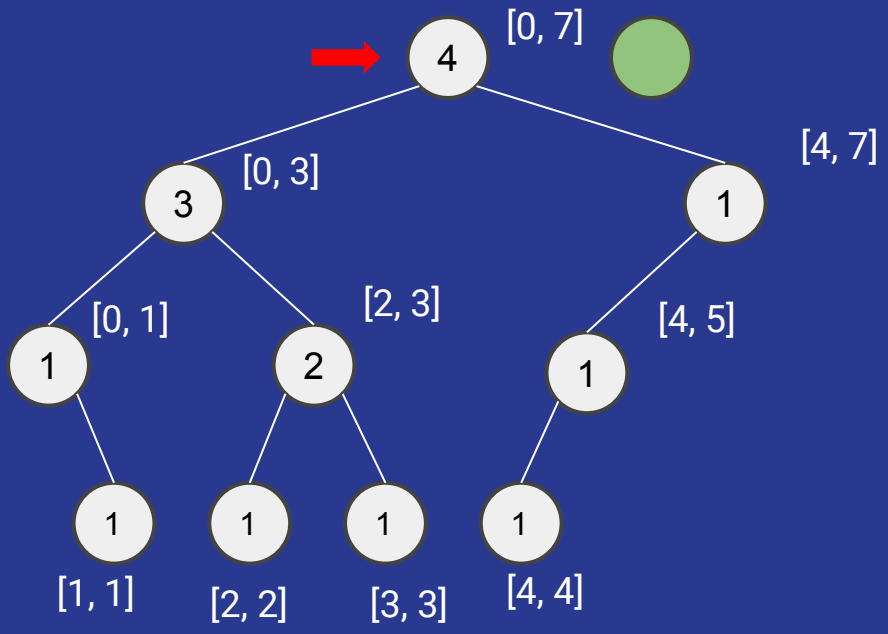
Implicit Sets - Split

split(2)



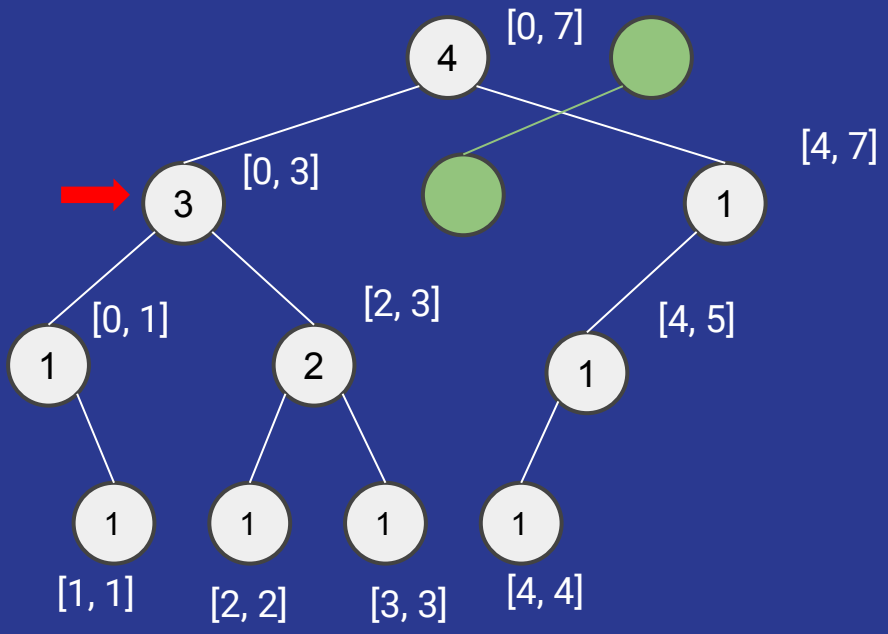
Implicit Sets - Split

split(2)



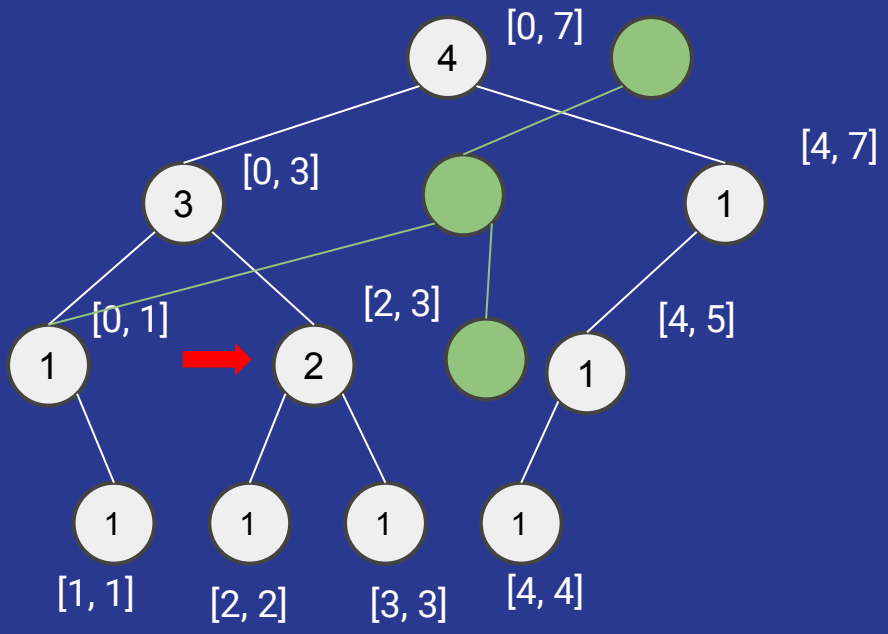
Implicit Sets - Split

split(2)



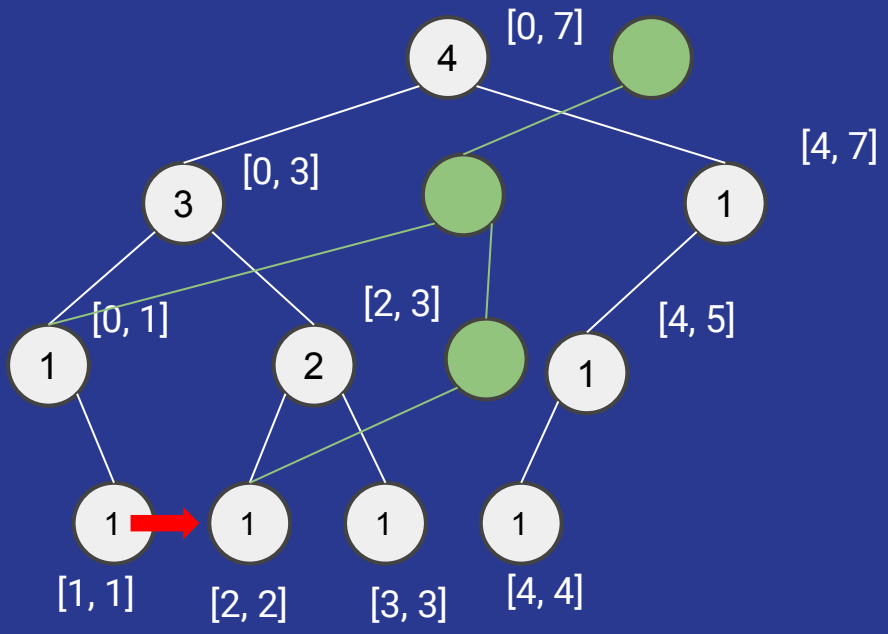
Implicit Sets - Split

split(2)



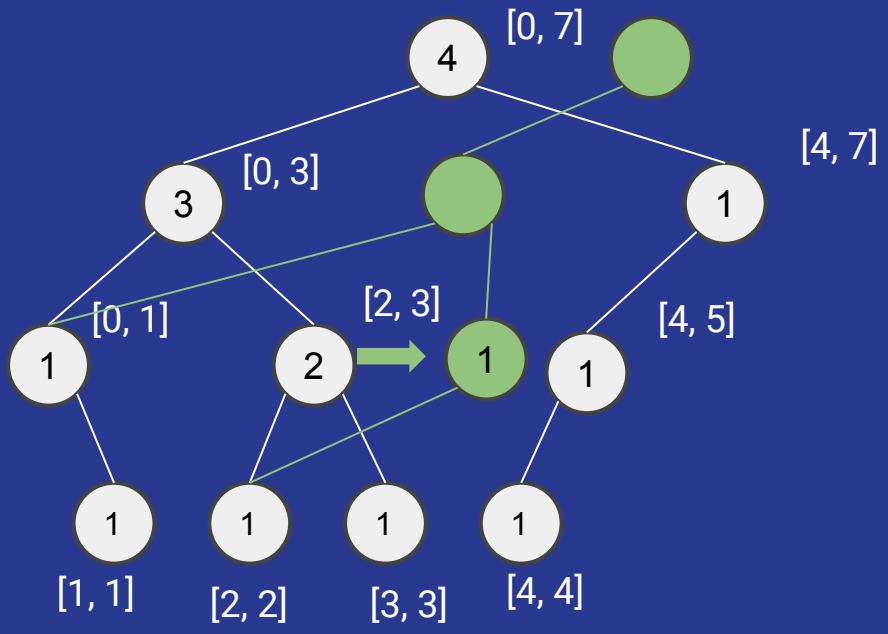
Implicit Sets - Split

split(2)



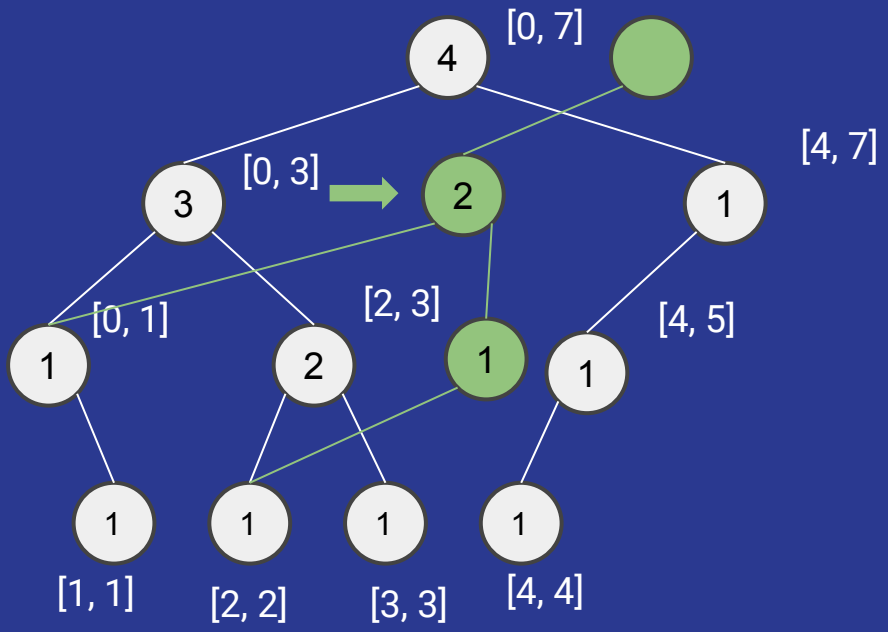
Implicit Sets - Split

split(2)



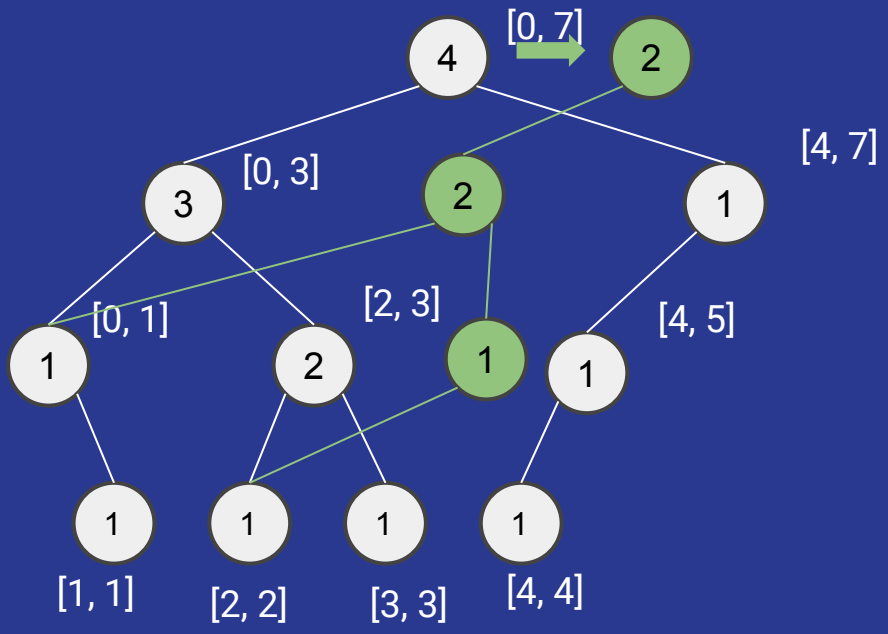
Implicit Sets - Split

split(2)



Implicit Sets - Split

split(2)

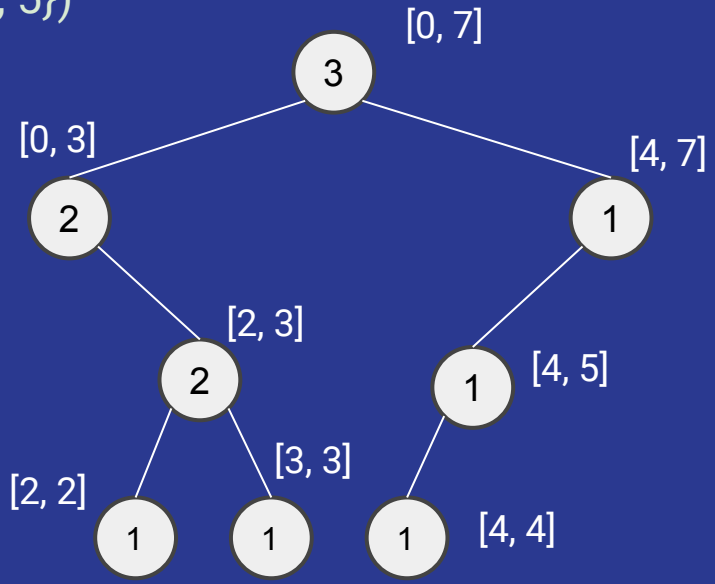


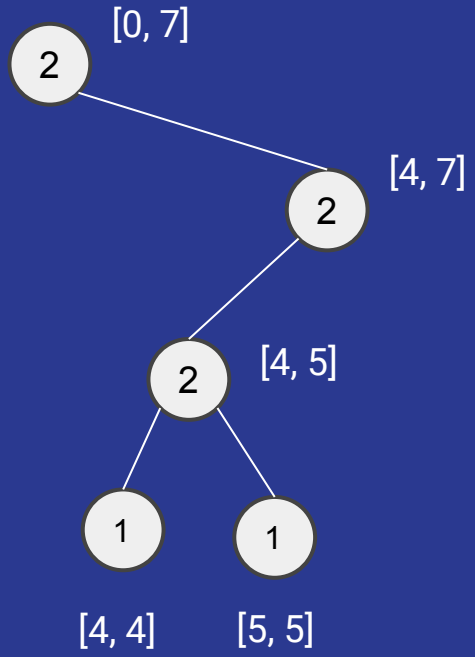
Implicit Sets - Split

O(lgN) novos nós

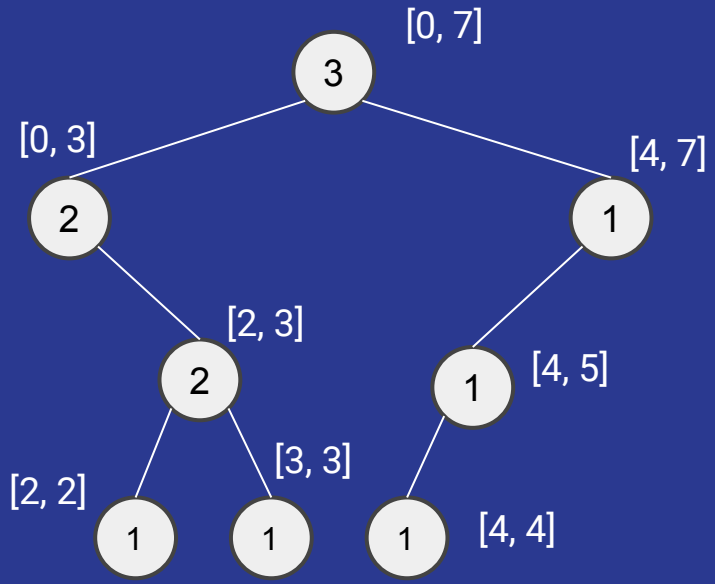
Implicit Sets

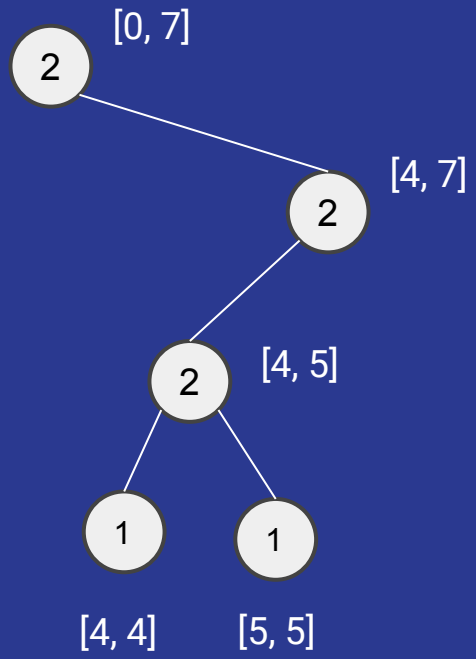
Merge({2, 3, 4}, {4, 5})





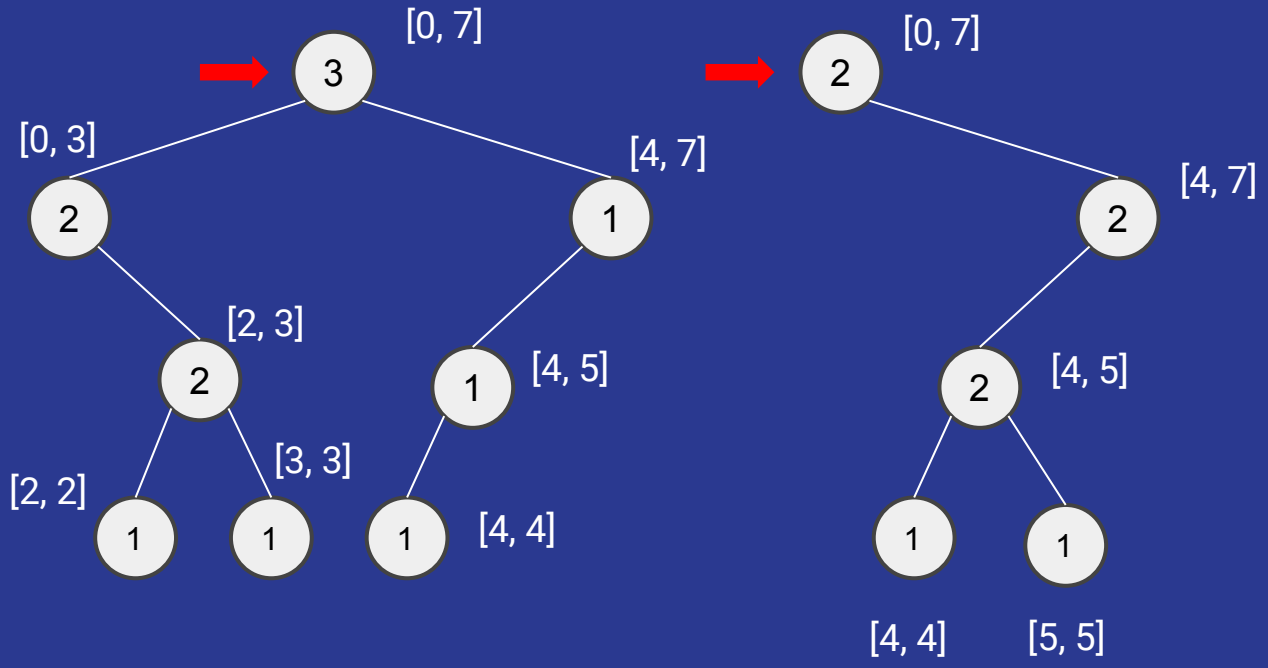
Implicit Sets - Merge



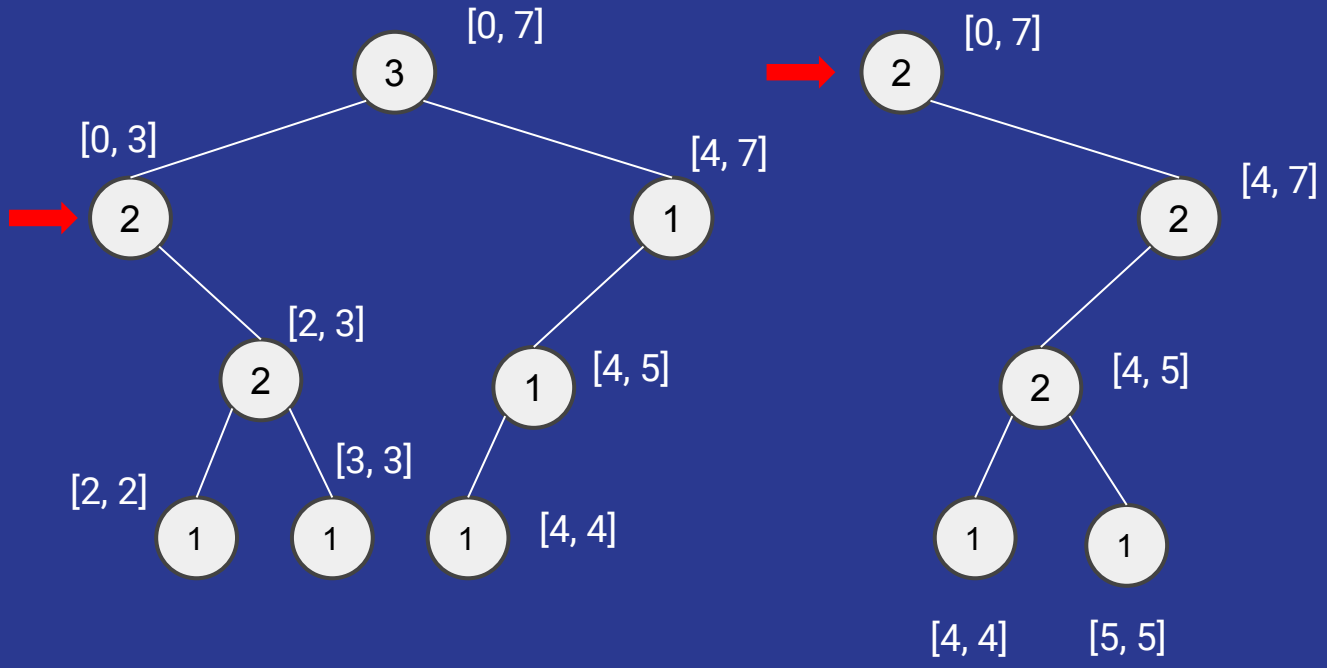


Só atravessa nós que estão presentes em ambos

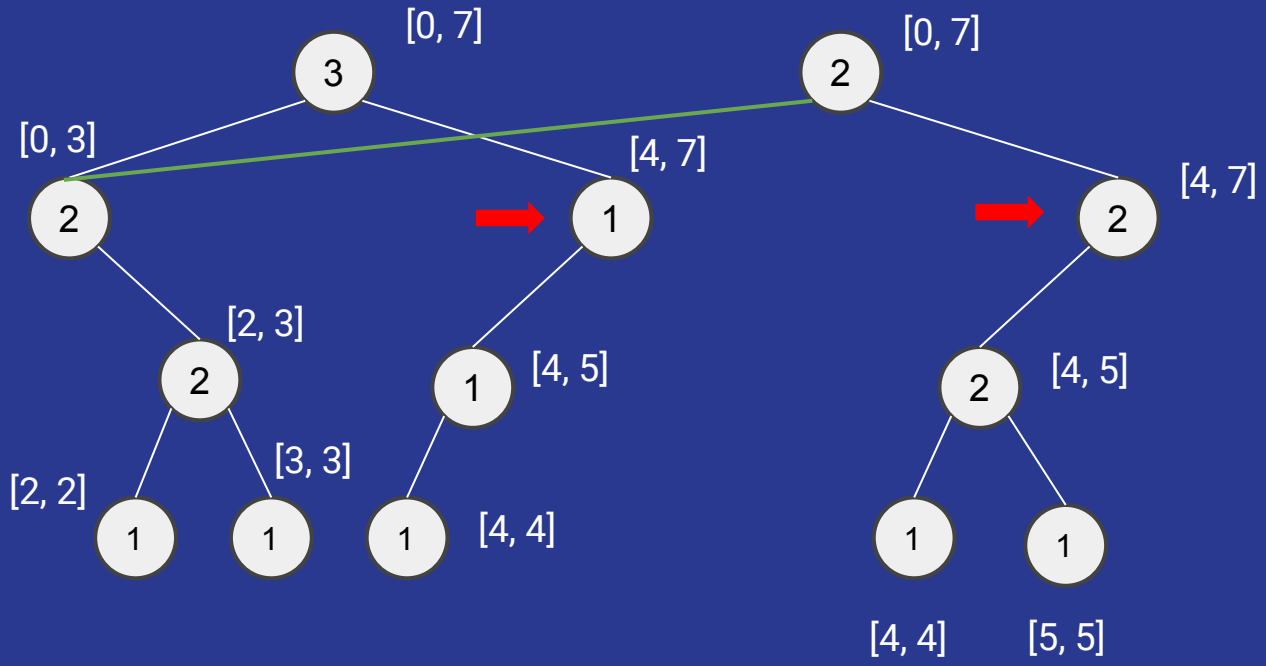
Implicit Sets - Merge



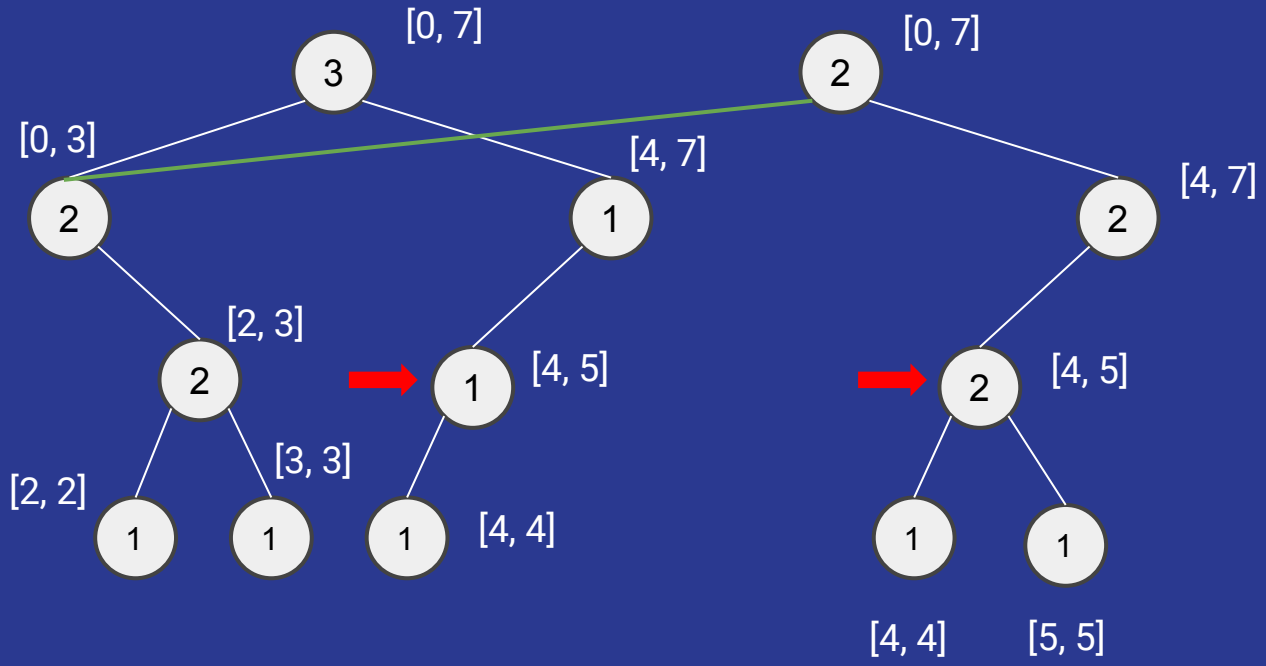
Implicit Sets - Merge



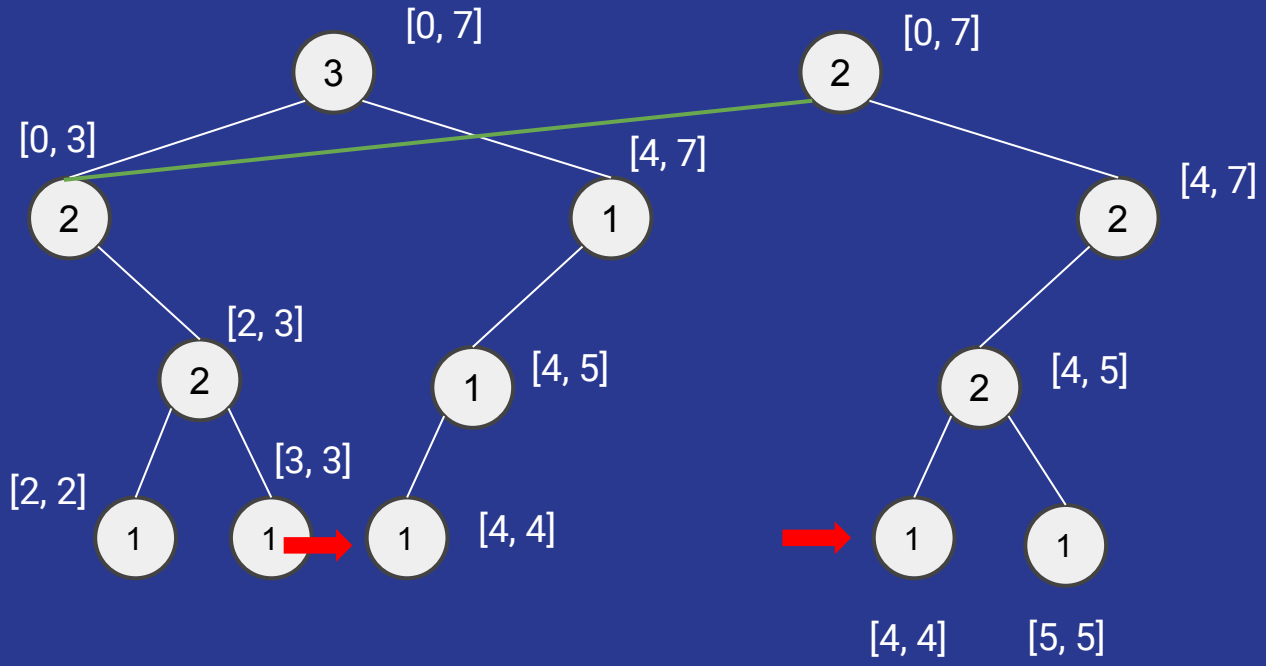
Implicit Sets - Merge



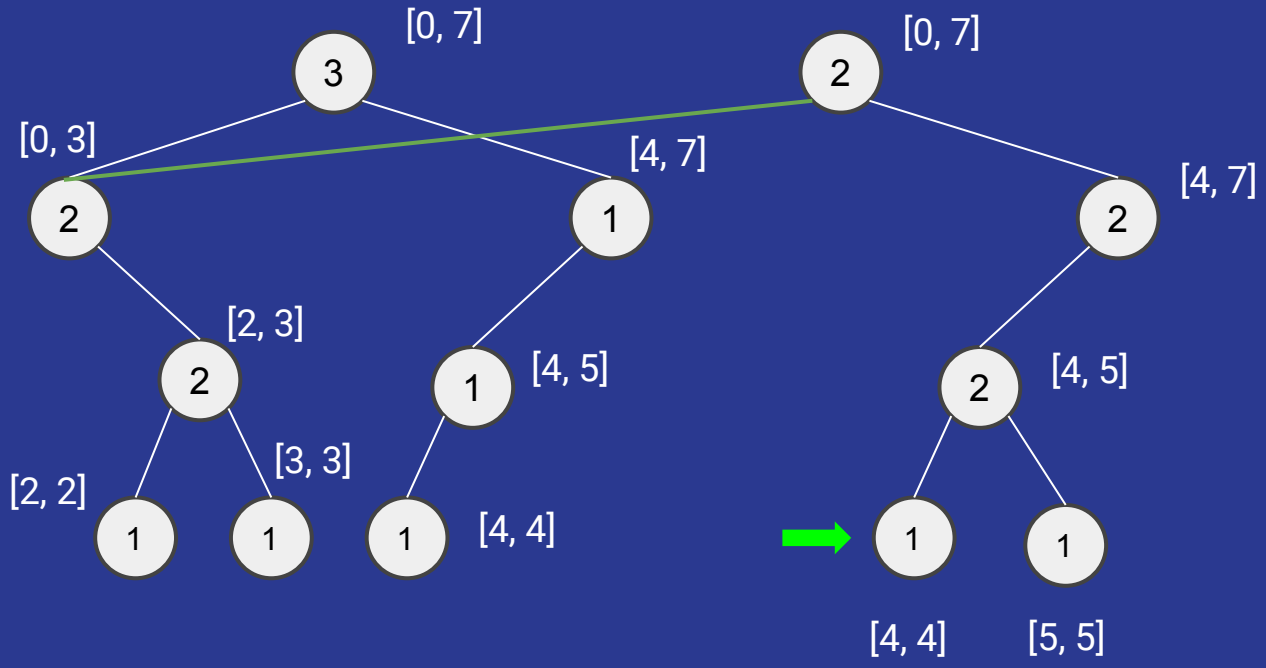
Implicit Sets - Merge



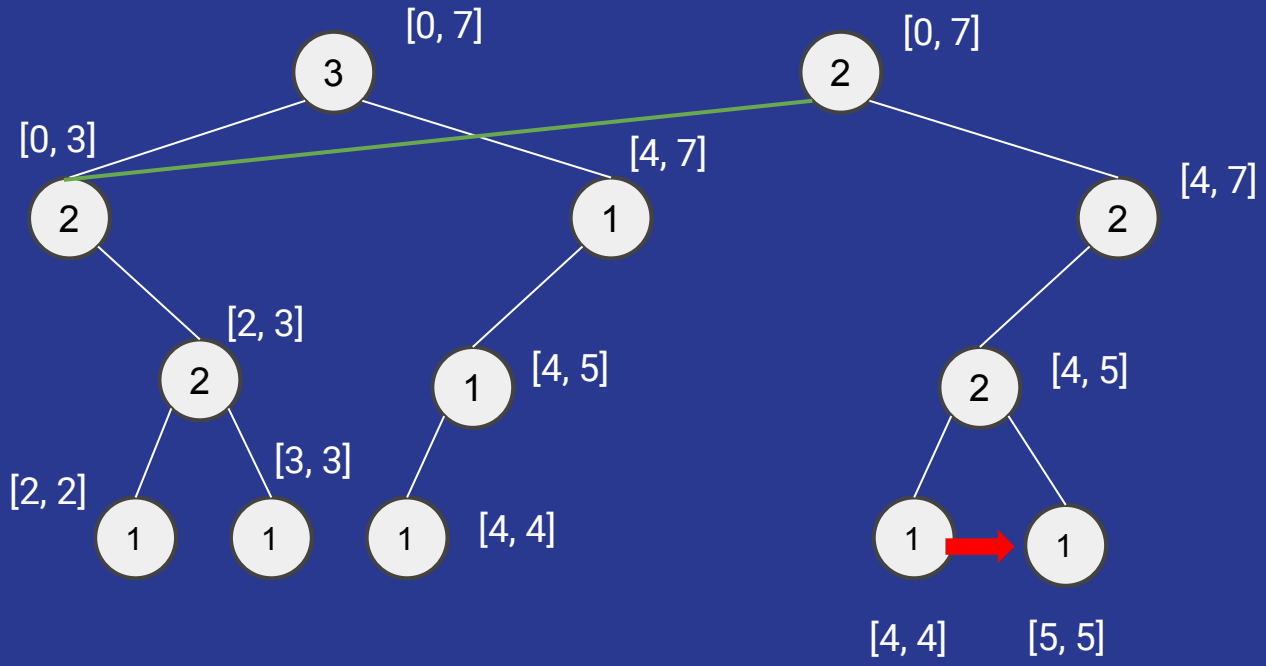
Implicit Sets - Merge



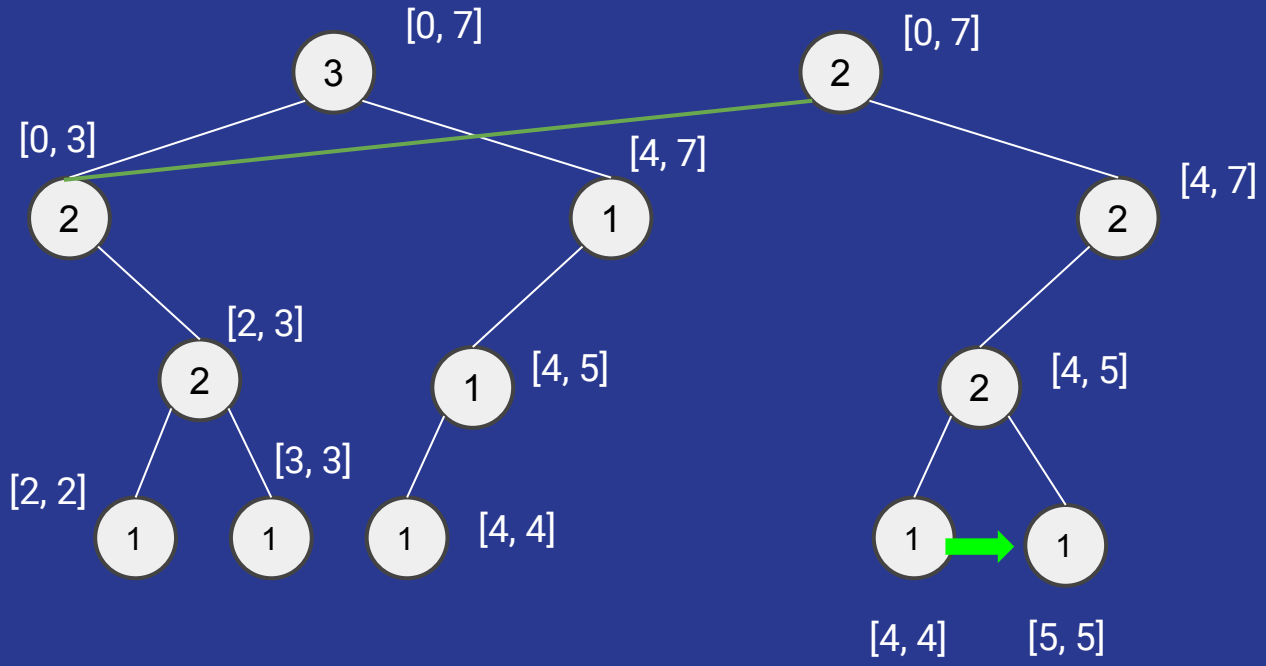
Implicit Sets - Merge



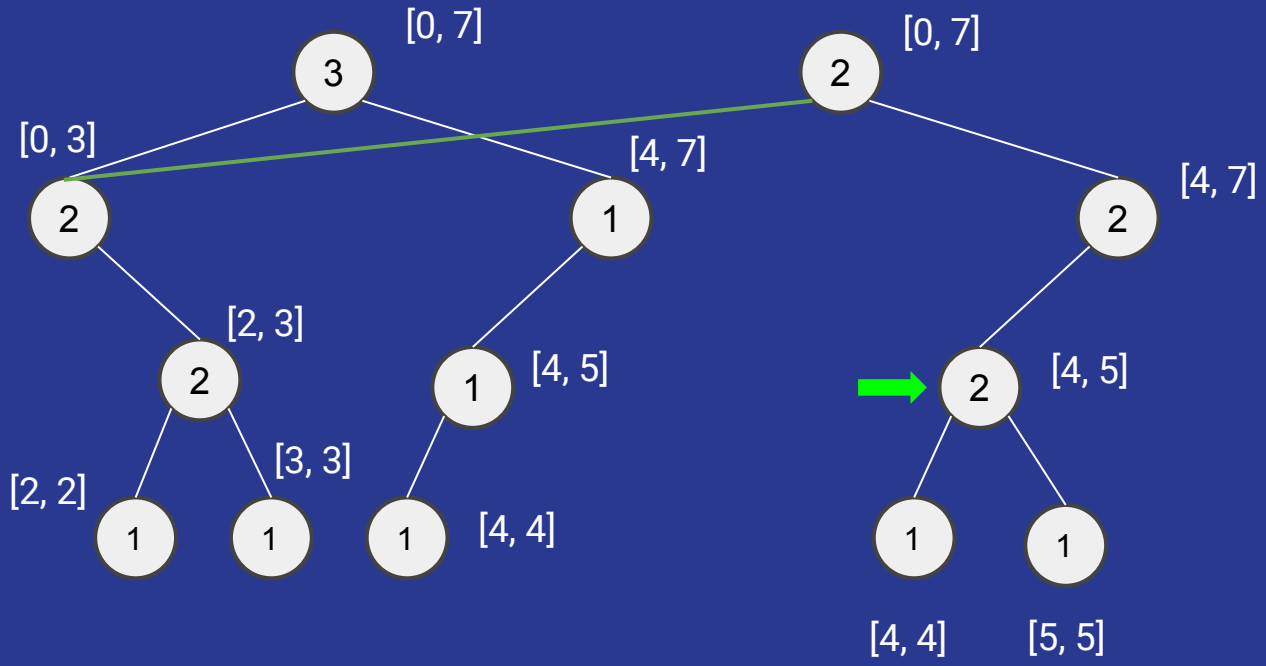
Implicit Sets - Merge



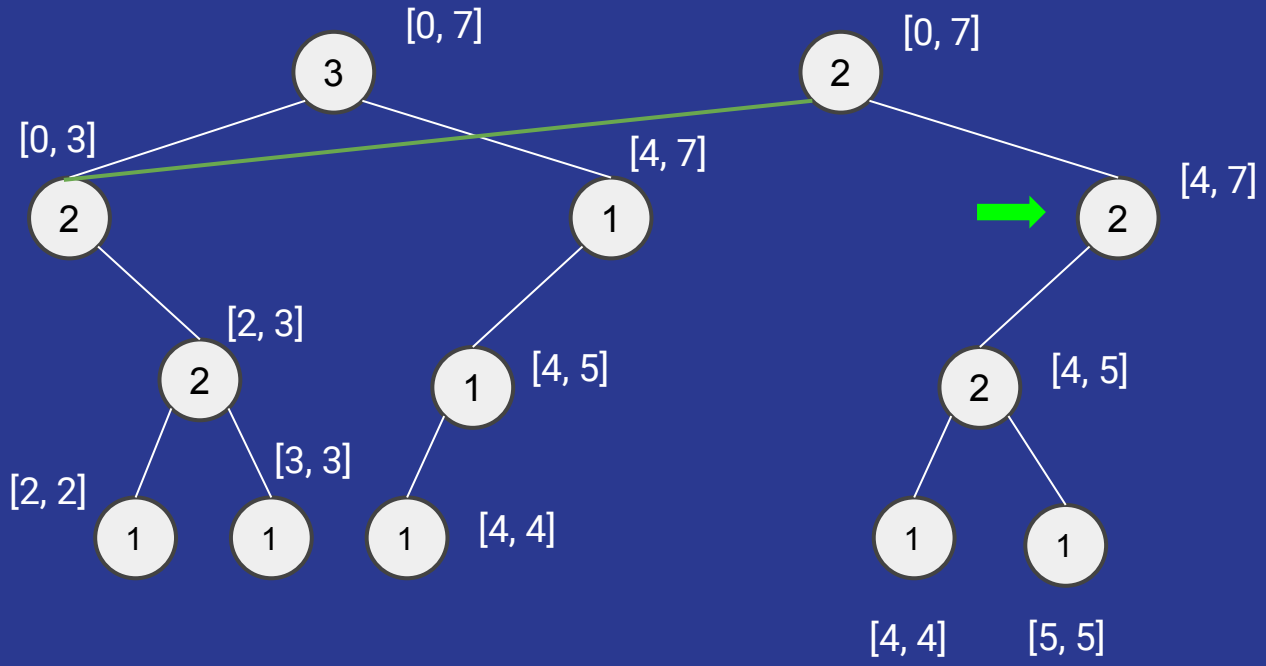
Implicit Sets - Merge



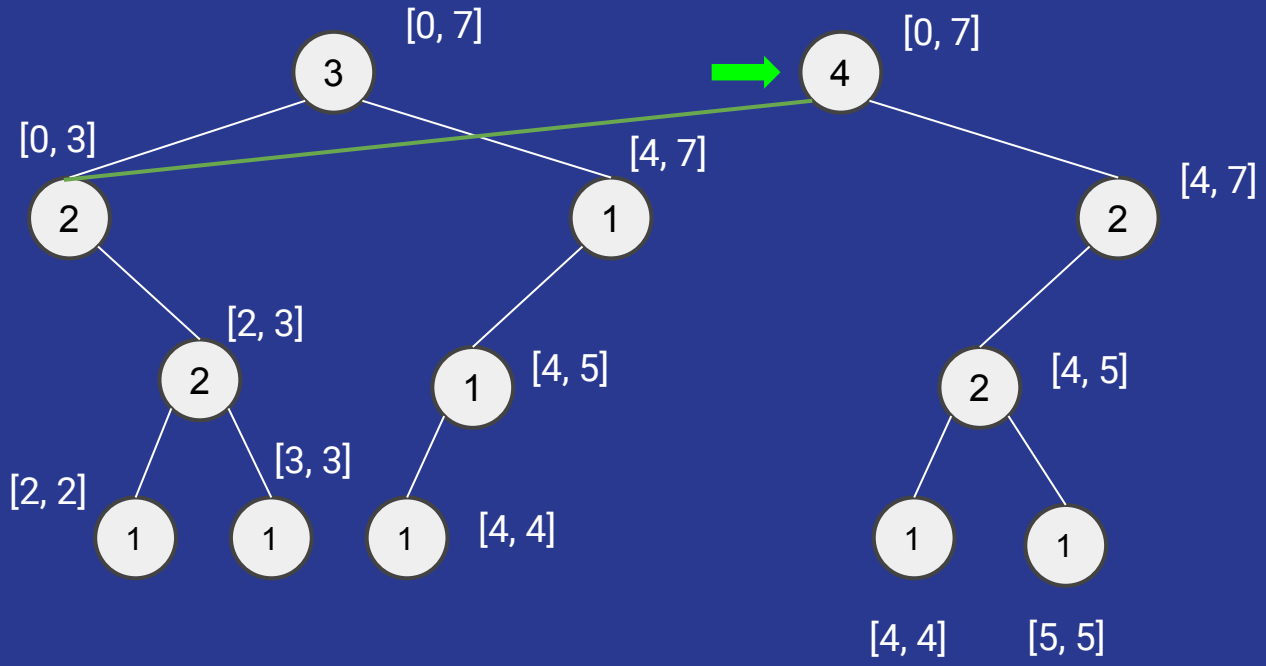
Implicit Sets - Merge



Implicit Sets - Merge



Implicit Sets - Merge



Implicit Sets - Merge

O pior caso pode ser O(N)

Mas cada par de nós percorrido descarta um

Implicit Sets - Implementação

https://github.com/lucasturci/Competitive_Programming/blob/master/Segtree/ImplicitSet/prog.cpp

Fontes

- https://www.youtube.com/watch?v=T0yzrZL1py0&t=1368s
- https://codeforces.com/blog/entry/49446
- https://discuss.codechef.com/t/difval-editorial/68523