Prioritetskøer



IN2010 – Uke 5

Prioritetskø

Abstrakt datatype – Kan implementeres på forskjellige måter, med forskjellige datastrukturer

Insert(e)

RemoveMin( )

Prioritetskø - Implementasjon

- Usortert lenket liste: O(1) på insert, O(n) på removeMin
- Sortert lenket liste: O(n) på insert, O(1) på removeMin

- AVL trær: O(log(n)) på insert, O(log(n)) på removeMin

Vi skal lære om hvordan implementere med binære heaps!

Hva er binære heaps?

- Et balansert binært tre (Men ikke et balansert binært **søke**tre)

Annerledes enn AVL trær:

AVL:
Venstre barn er lavere enn noden
Høyre barn er høyere enn noden

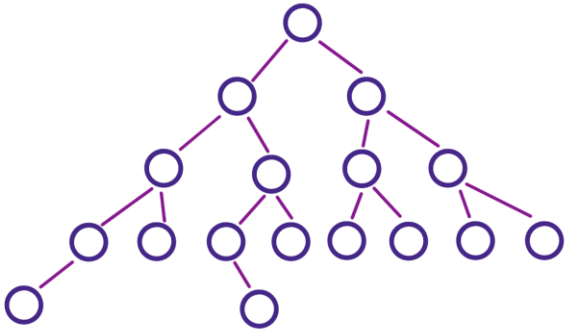
Heaps:
Alle noder er større (eller lik) foreldernoden sin

Heaps må også være komplette!

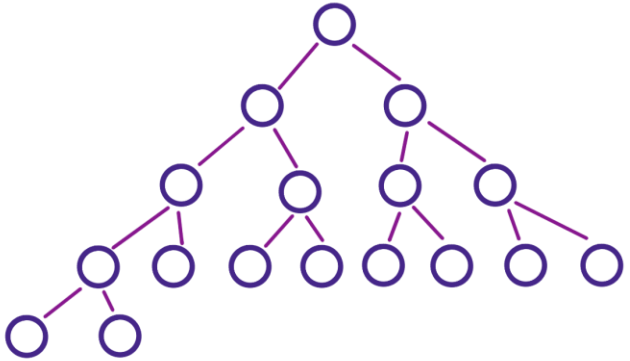
Hva er et komplett tre???

- Et tre som fylles opp fra venstre til høyre

Ikke valid:

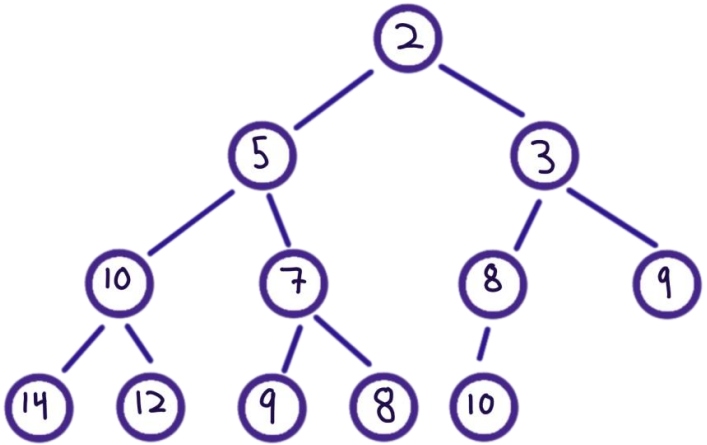


Valid:



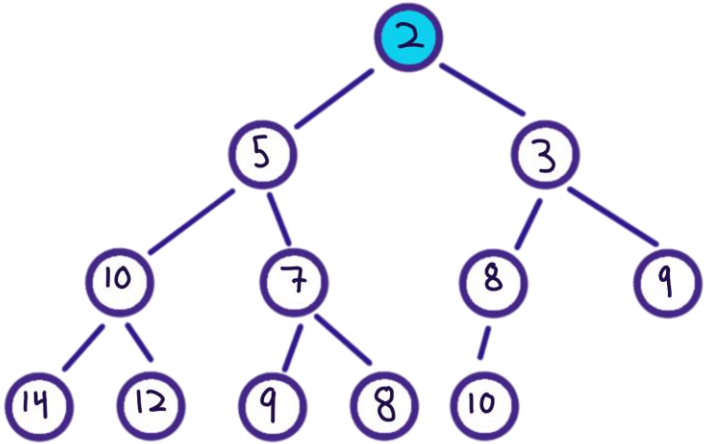
Heaps i praksis

- Vi konstruerer en heap ved å bruke et array

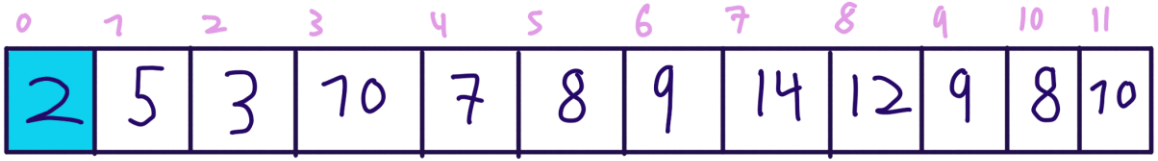


Heaps i praksis

- Vi konstruerer en heap ved å bruke et array

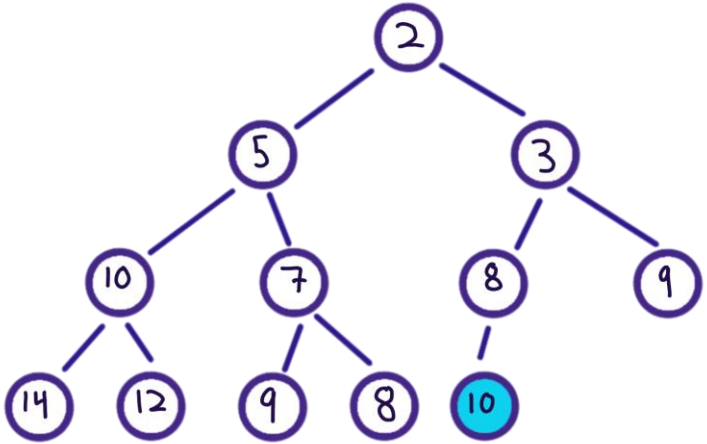


A[0] = rotnoden

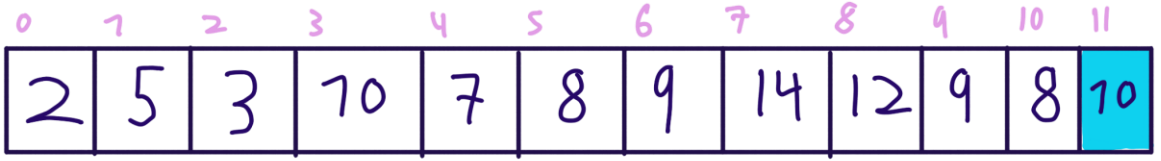


Heaps i praksis

- Vi konstruerer en heap ved å bruke et array



A[n-1] = Siste noden i treet



Tre veldig viktige hjelpemetoder

```
1 Procedure ParentOf(i)
2   return ⌊(i−1)/2⌋
```

```
1 Procedure LeftOf(i)
2   return 2 · i + 1
```

```
1 Procedure RightOf(i)
2   return 2 · i + 2
```

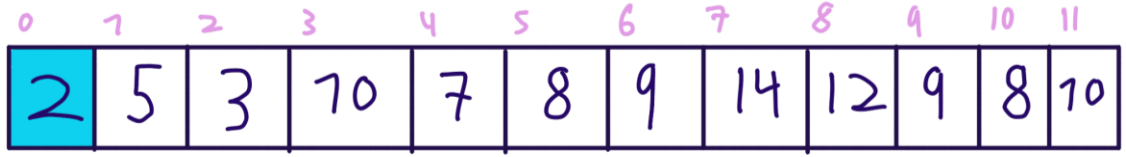
Tre veldig viktige hjelpemetoder

```
1 Procedure ParentOf(i)
2 |  return ⌊(i−1)/2⌋
```

```
1 Procedure LeftOf(i)
2 |  return 2 · i + 1
```

```
1 Procedure RightOf(i)
2 |  return 2 · i + 2
```

a) Hva er forelderen til tallet ‘12’?
b) Hva er det høyre barnet til tallet ‘3’?
c) Hva er det venstre barnet til tallet ‘7’



Innsetting i heaps

- Setter inn det nye elementet ved posisjon A[n]
- ‘Shuffler’ det nye elementet opp til den har en foreldrenode som er mindre enn den

ALGORITHM: INNSETTING I HEAP

Input: Et array A som representerer en heap med n elementer, og et element x
Output: Et array som representerer en heap, som inneholder x

```
1 Procedure Insert(A, x)
2     A[n] ← x
3     i ← n
4     while 0 < i and A[i] < A[ParentOf(i)] do
5         swap A[i] and A[ParentOf(i)]
6         i ← ParentOf(i)
```

Sletting i heaps

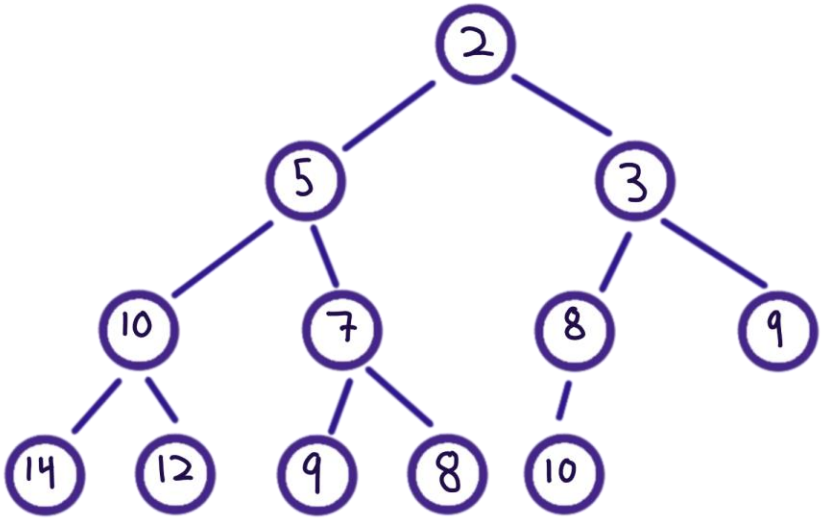
ALGORITHM: FJERNING AV MINSTE ELEMENT FRA HEAP

Input: Et array A som representerer en ikke-tom heap med n elementer
Output: Det minste elementet fjernes og returneres

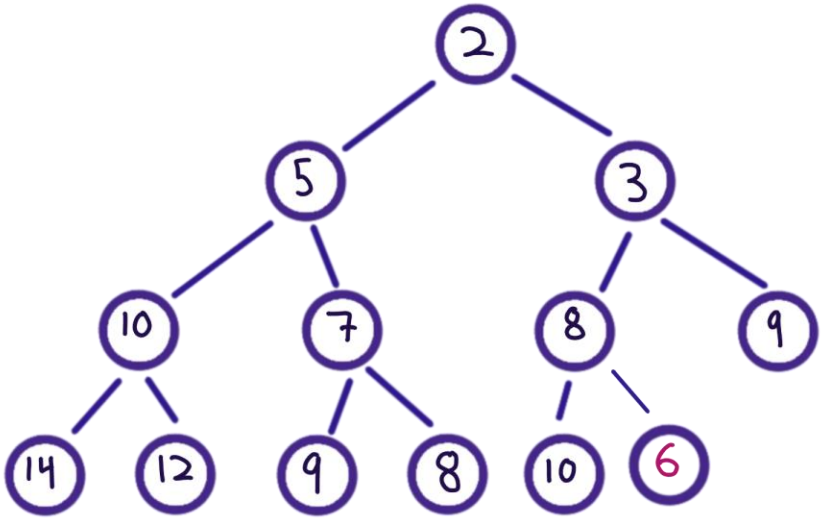
```
1  Procedure RemoveMin(A)
2      x ← A[0]
3      A[0] ← A[n − 1]
4      i ← 0
5      while LeftOf(i) < n − 1 do
6          j ← LeftOf(i)
7          if RightOf(i) < n − 1 and A[RightOf(i)] < A[j] then
8              j ← RightOf(i)
9          if A[i] ≤ A[j] then
10             return x
11         swap A[i] and A[j]
12         i ← j
13     return x
```

- Fjerner roten (som vi vet er minst)
- ‘Oppgraderer’ siste node (A[n-1]) til den nye roten
- ‘Shuffle’ den nye roten nedover til den har barn som er større enn seg selv

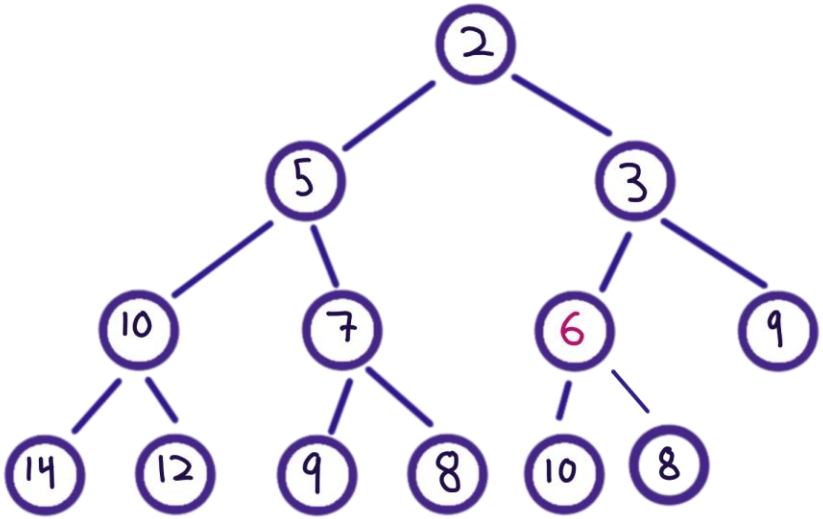
Hva skjer dersom vi setter inn tallet ‘6’ i heapen?



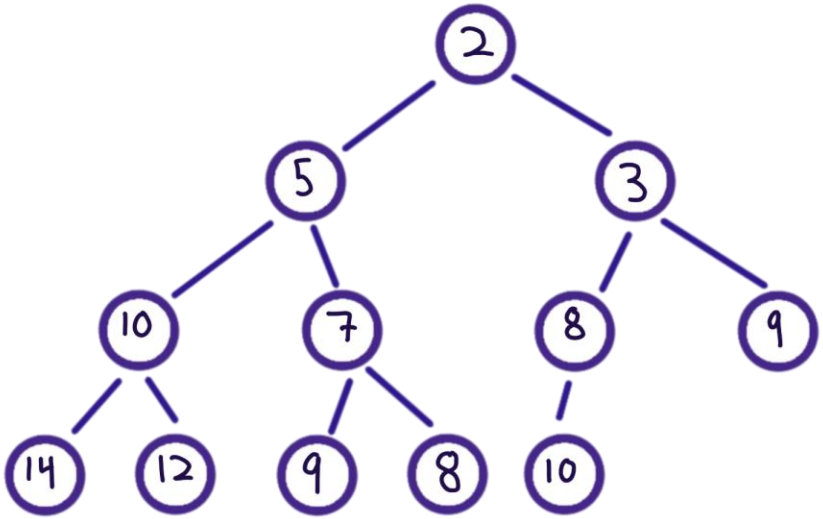
Hva skjer dersom vi setter inn tallet ‘6’ i heapen?



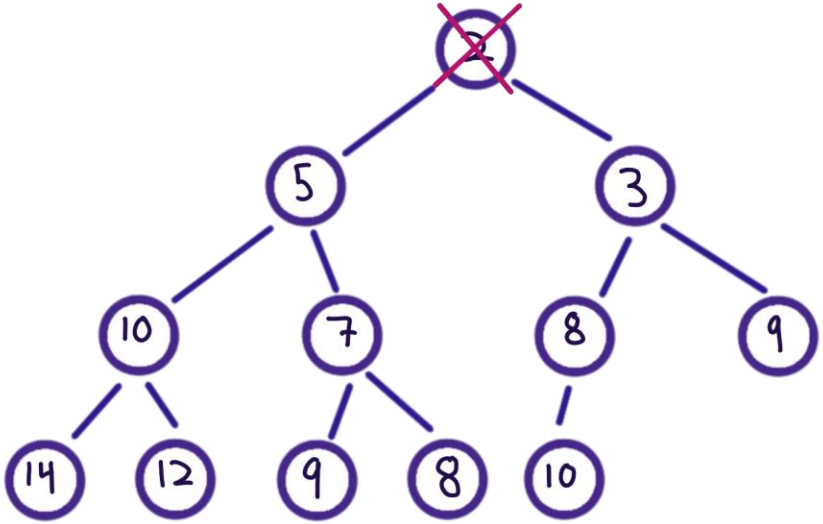
Hva skjer dersom vi setter inn tallet ‘6’ i heapen?



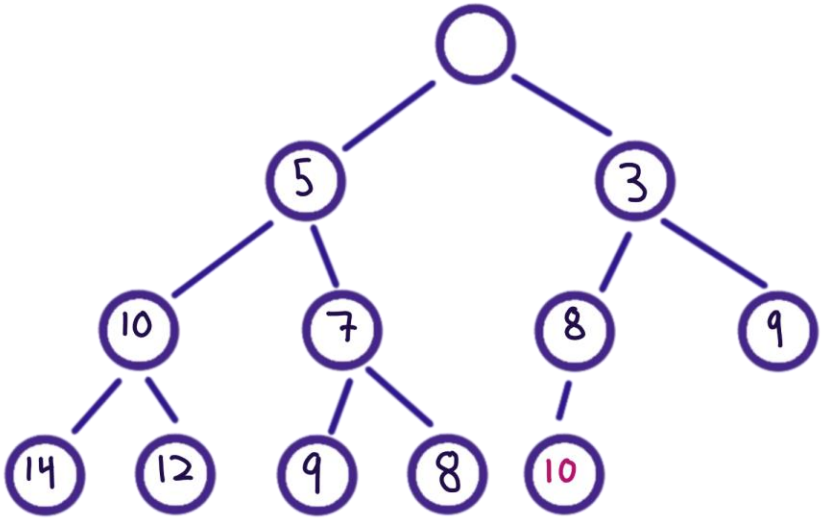
Hva skjer dersom vi gjøre removeMin() på heapen?



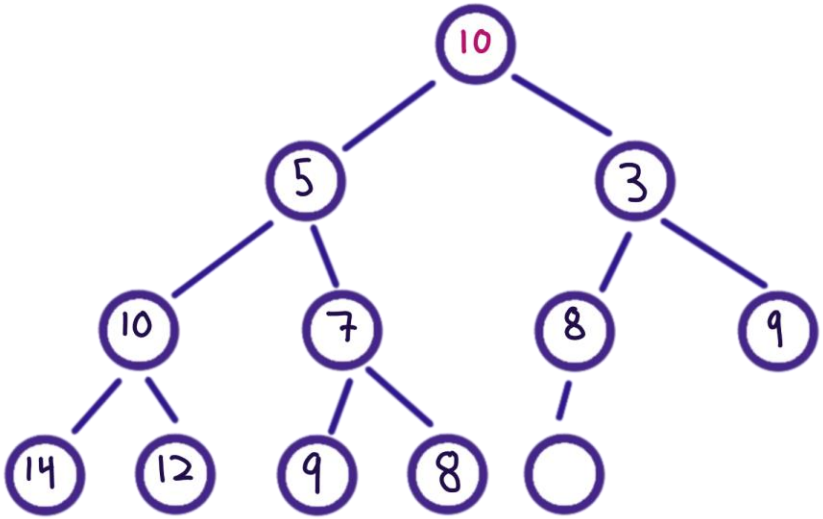
Hva skjer dersom vi gjøre removeMin() på heapen?



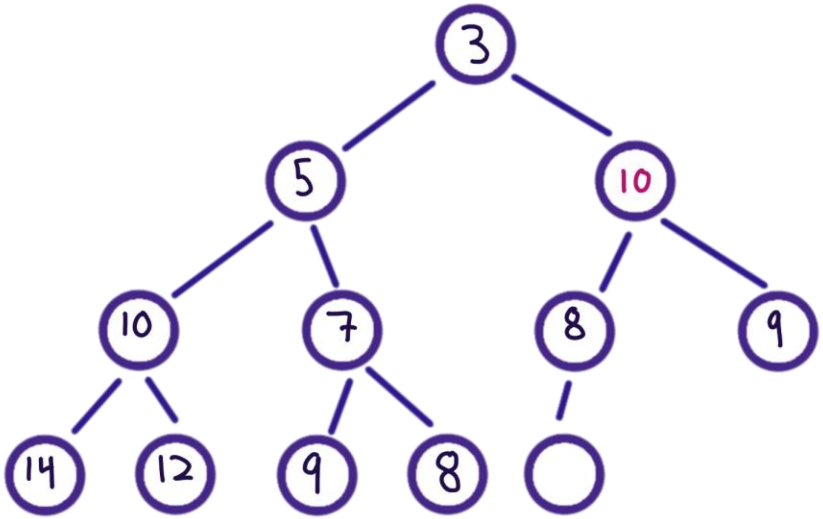
Hva skjer dersom vi gjøre removeMin() på heapen?



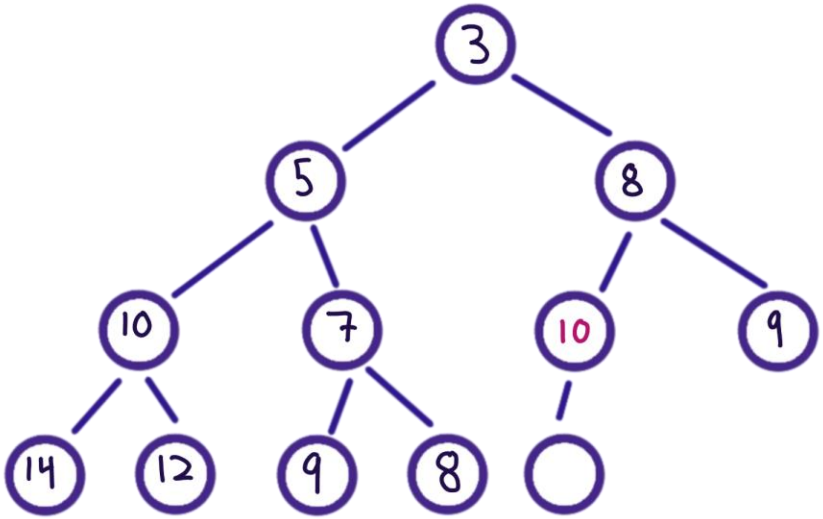
Hva skjer dersom vi gjøre removeMin() på heapen?



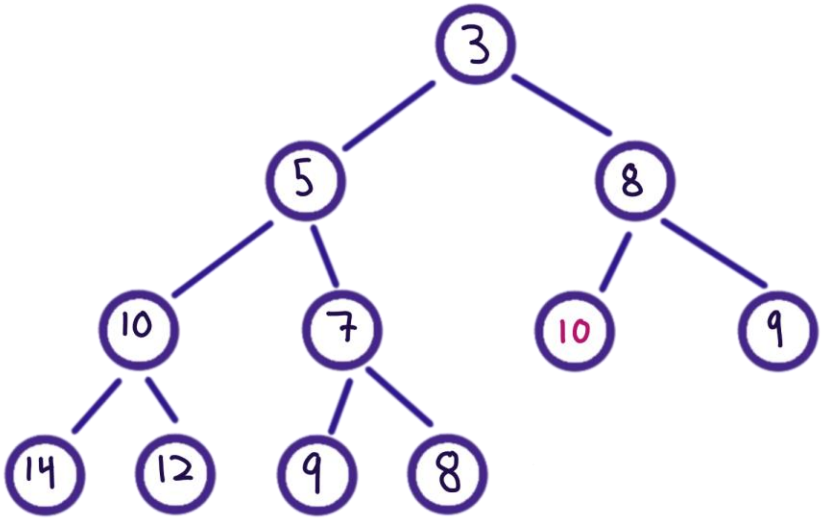
Hva skjer dersom vi gjøre removeMin() på heapen?



Hva skjer dersom vi gjøre removeMin() på heapen?



Hva skjer dersom vi gjøre removeMin() på heapen?



Eksamensoppgave fra H22:

(b) En min-heap blir en max-heap ved å reversere arrayet.

Huffman-koding

Huffman

Problem: Vi vil representere så mange strenger/symboler som mulig med minst mulig plass

Løsning: Vi lager en frekvenstabell over alle bokstavene som dukker opp i strengen vi vil representere

- Da er bokstavene vi bruker mest representert mest færrest mulig bits

Ukas video – Huffman trær



https://youtu.be/JsTptu56GM8

Huffman

«Hello world!»



Huffman

«Hello world!»

L | 3

O | 2

H | 1

E | 1

W | 1

D | 1

R | 1

Huffman

«Hello world!»

D,R | 2

L | 3

O | 2

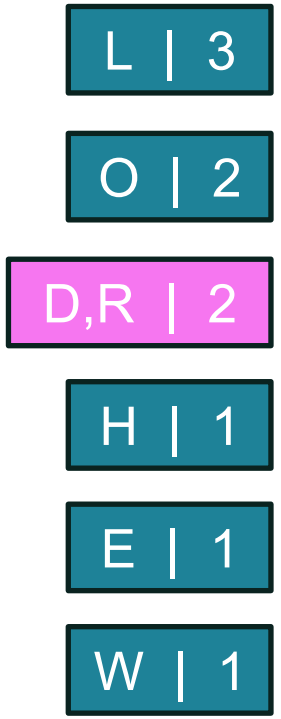
H | 1

E | 1

W | 1

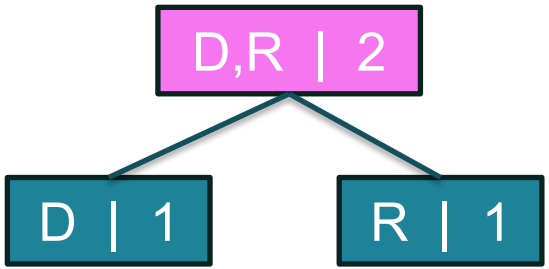
Huffman

«Hello world!»



Huffman

«Hello world!»



L | 3

O | 2

D,R | 2

H | 1

E | 1

W | 1

Huffman

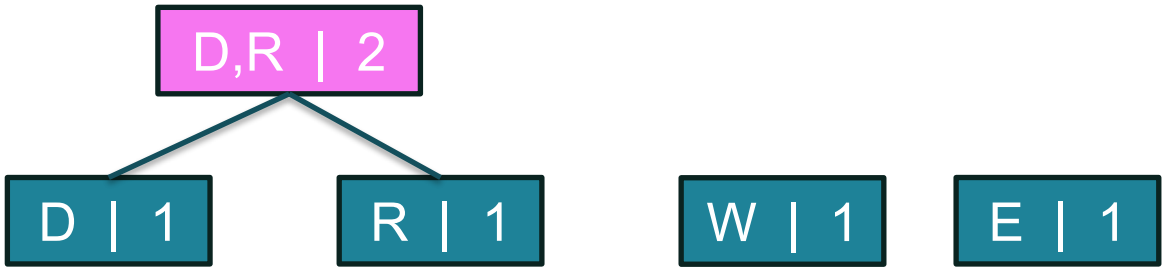
«Hello world!»

L | 3

O | 2

D,R | 2

H | 1



Huffman

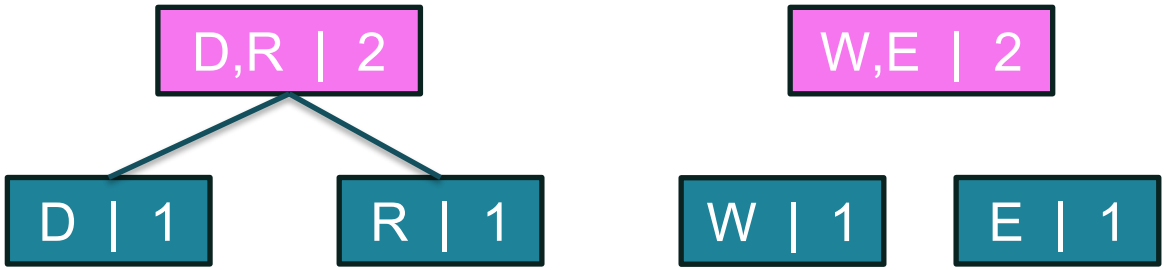
«Hello world!»

L | 3

O | 2

D,R | 2

H | 1



Huffman

«Hello world!»

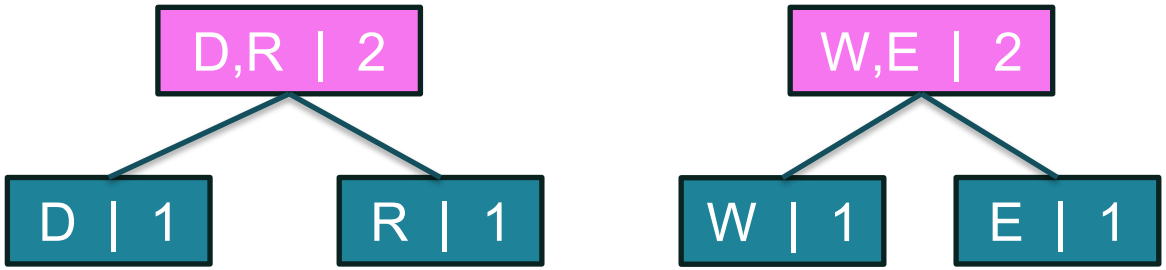
L | 3

O | 2

D,R | 2

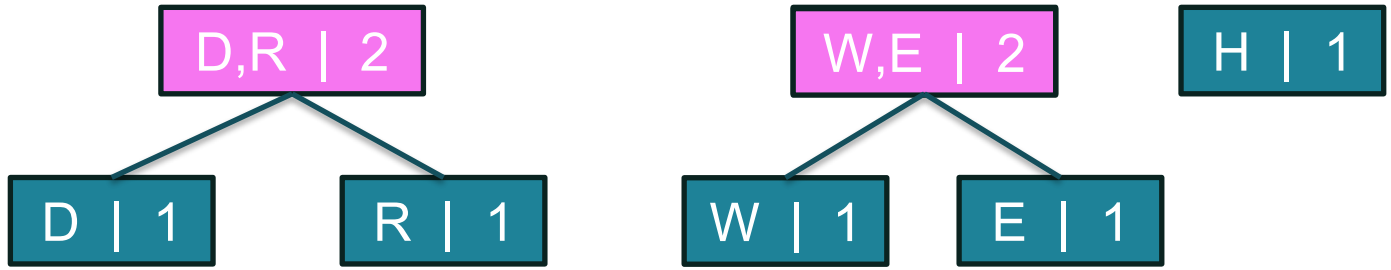
W,E | 2

H | 1



Huffman

«Hello world!»



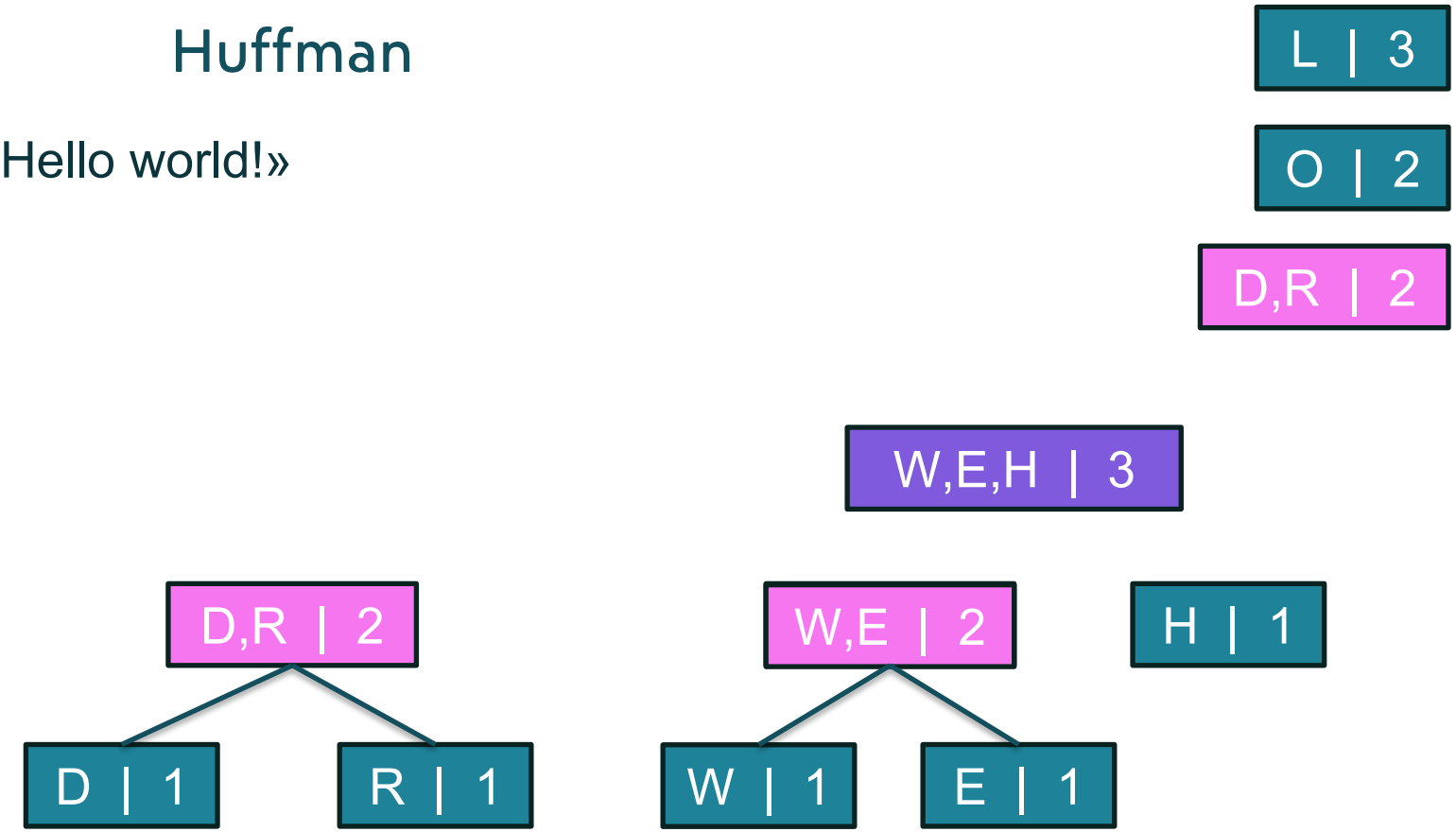
L | 3

O | 2

D,R | 2

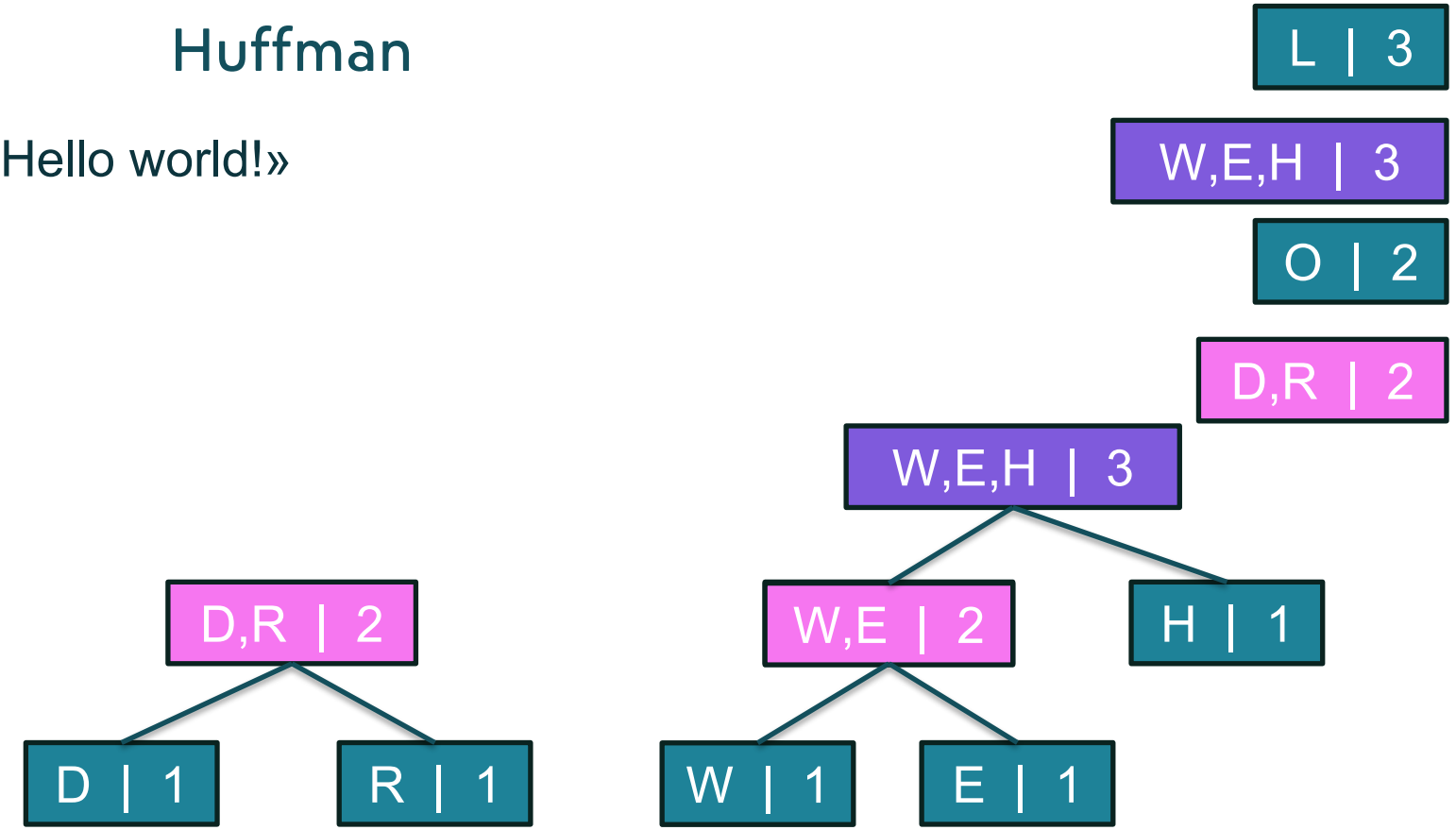
Huffman

«Hello world!»



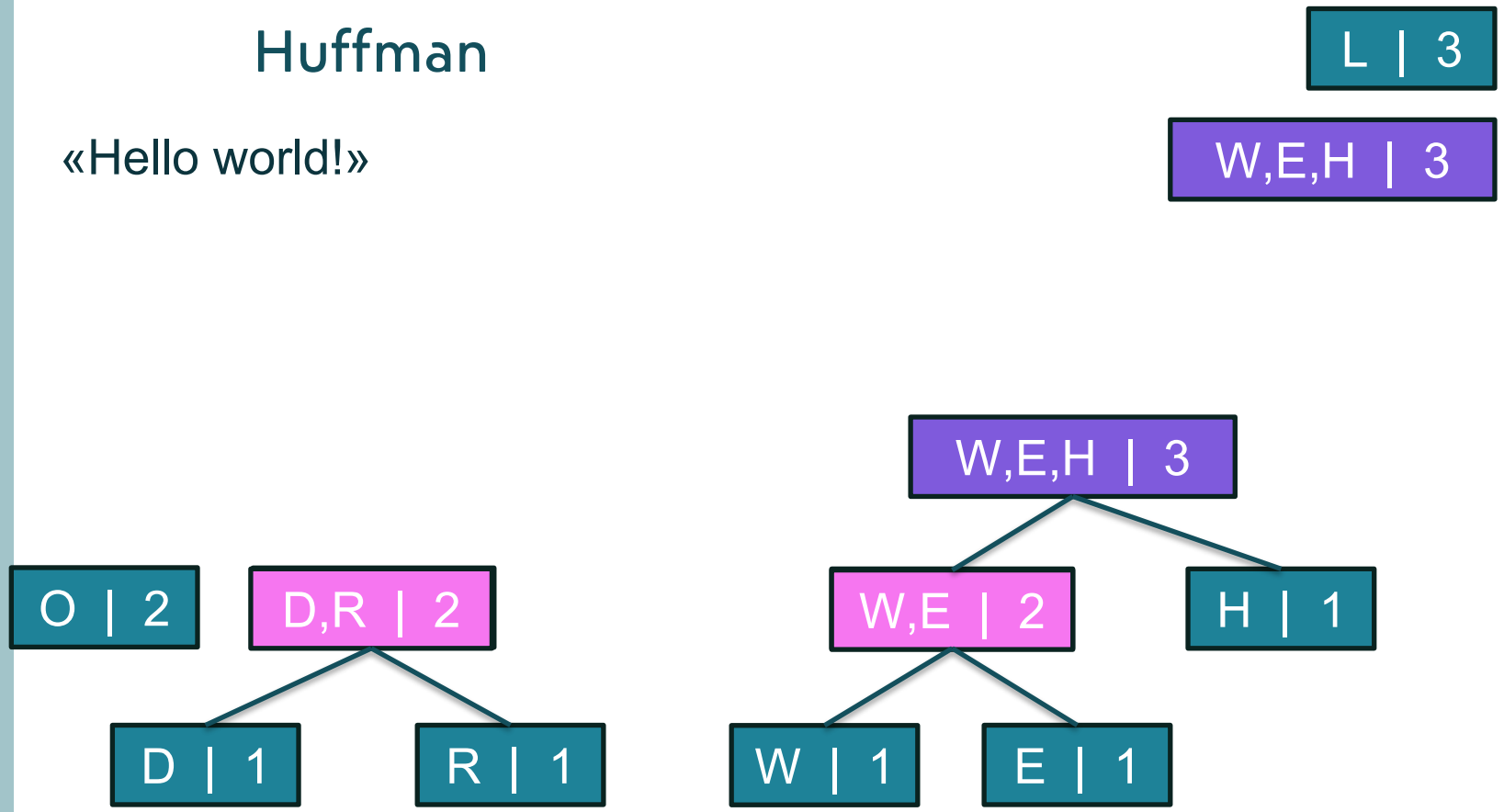
Huffman

«Hello world!»



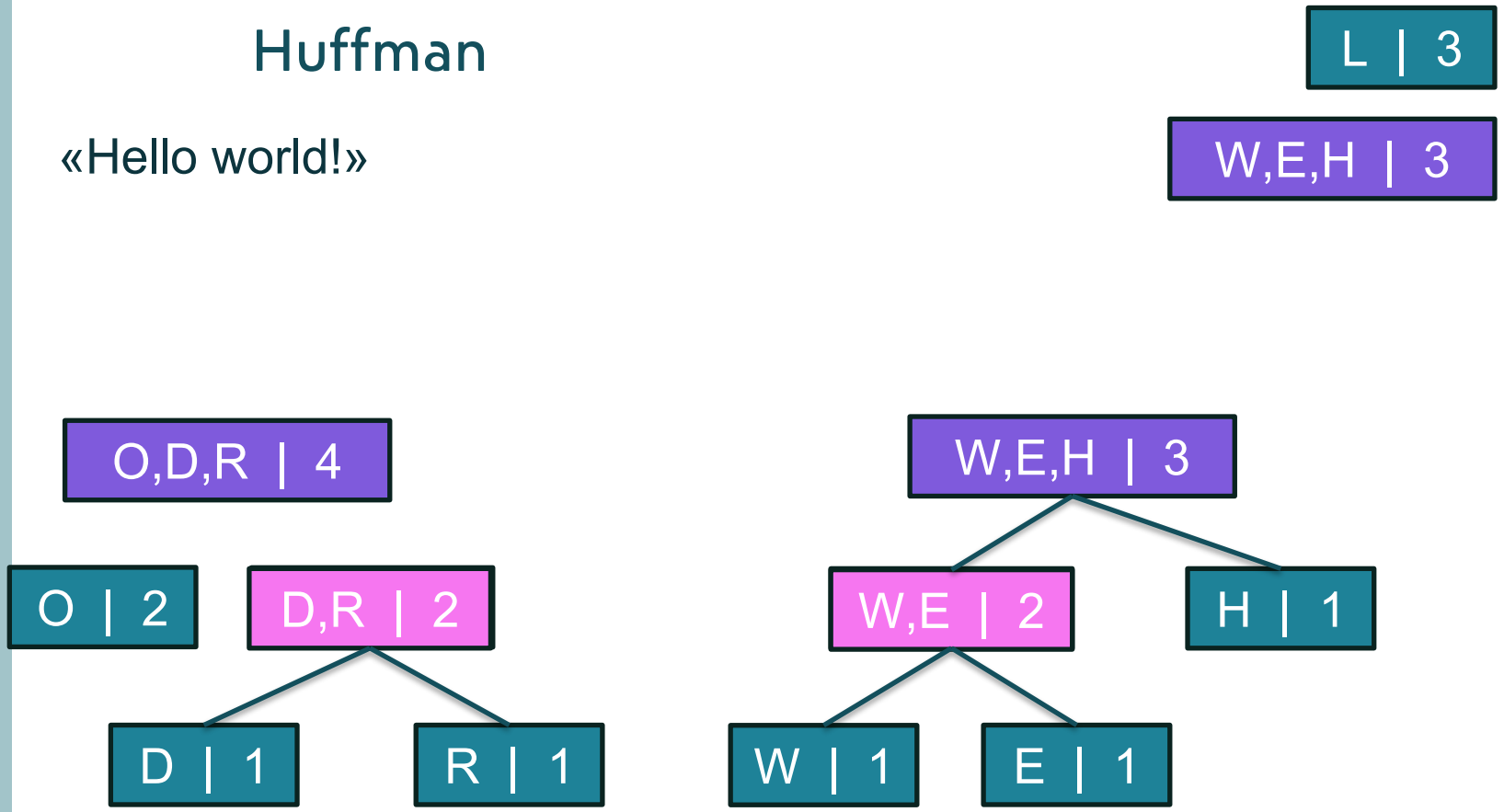
Huffman

«Hello world!»



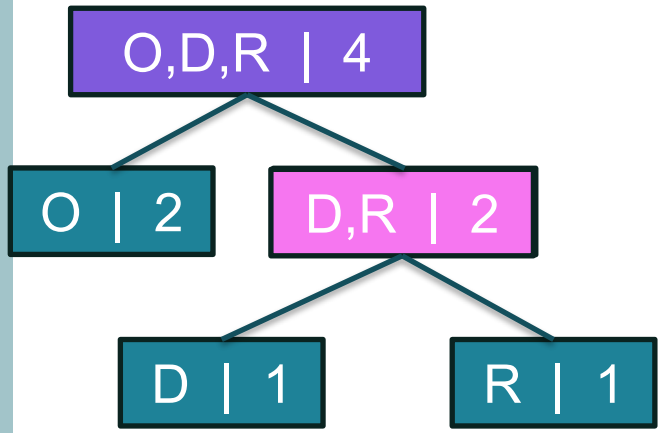
Huffman

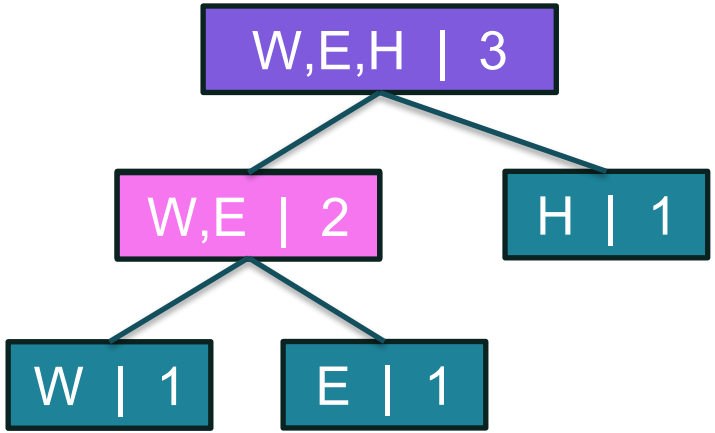
«Hello world!»



Huffman

«Hello world!»





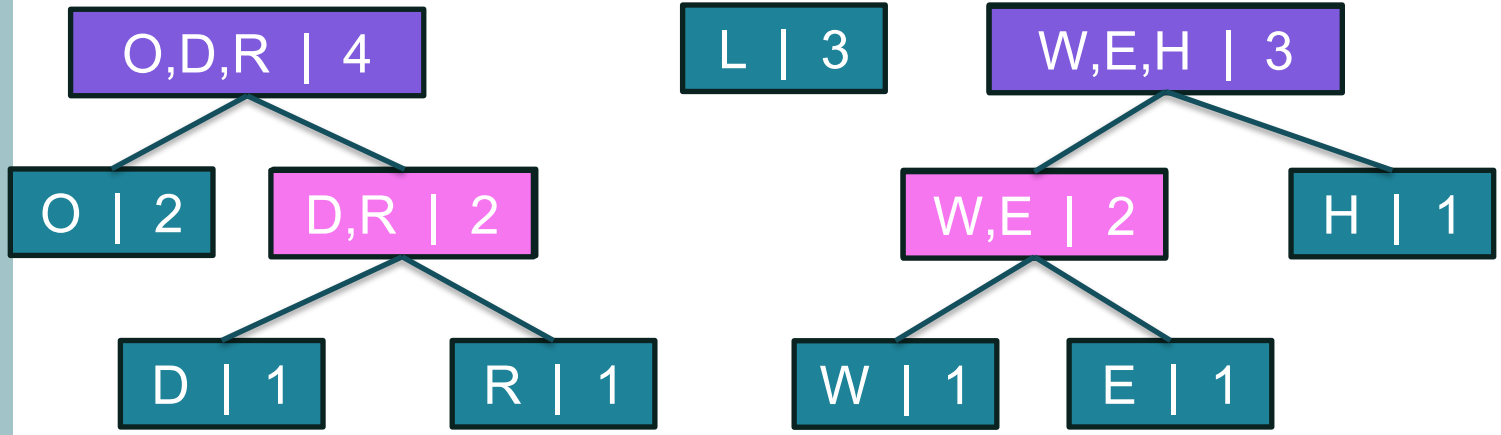
O,D,R | 4

L | 3

W,E,H | 3

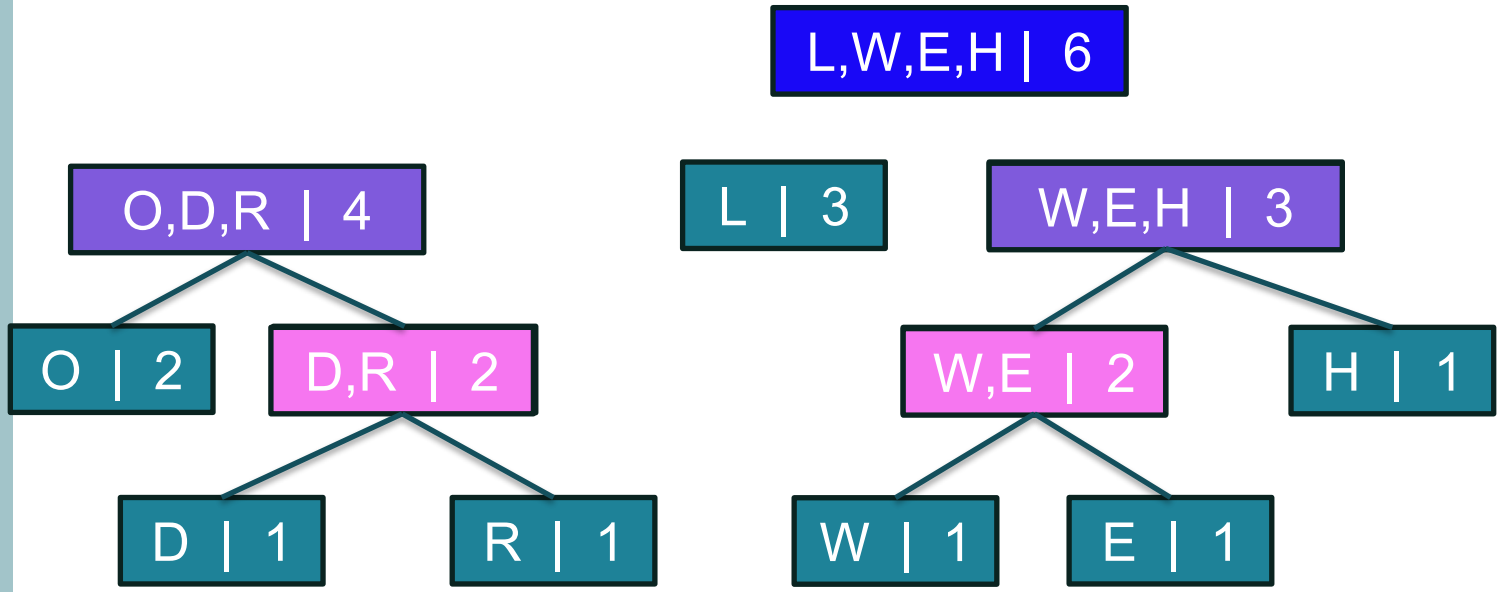
Huffman

«Hello world!»



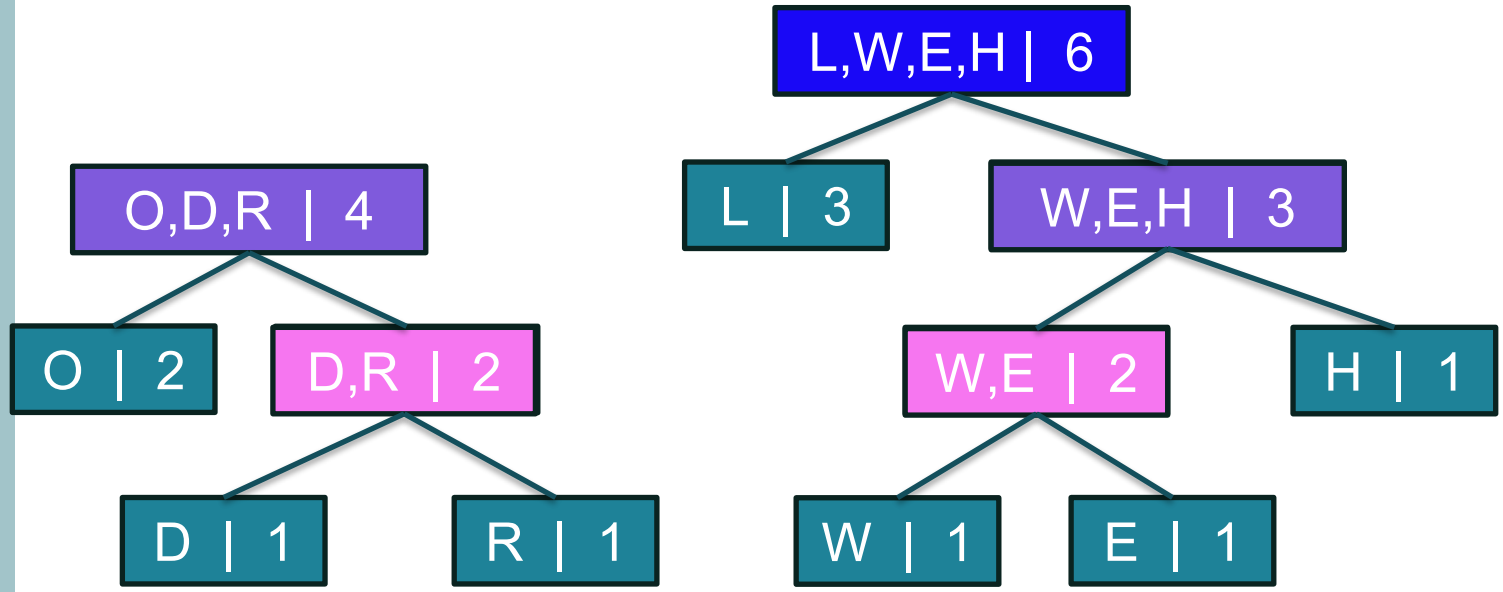
Huffman

«Hello world!»



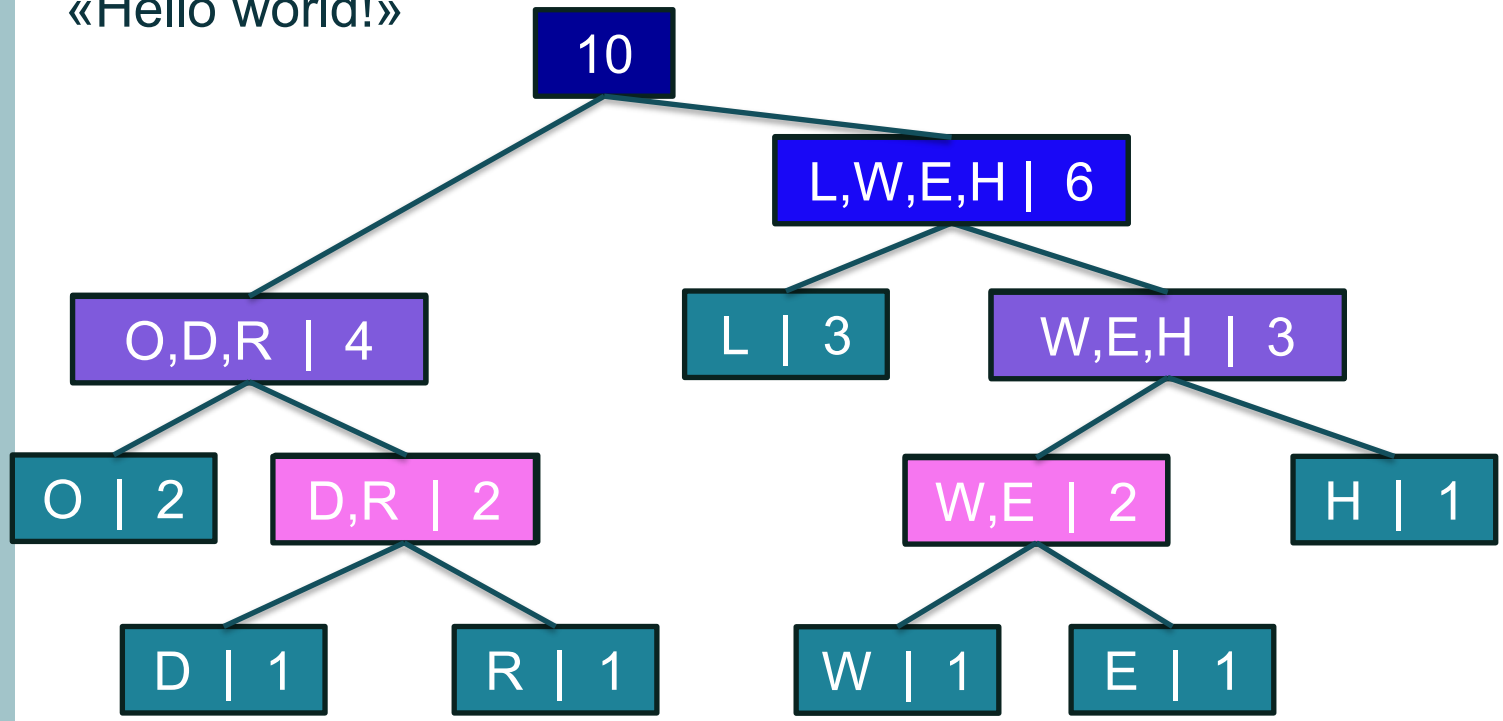
Huffman

«Hello world!»



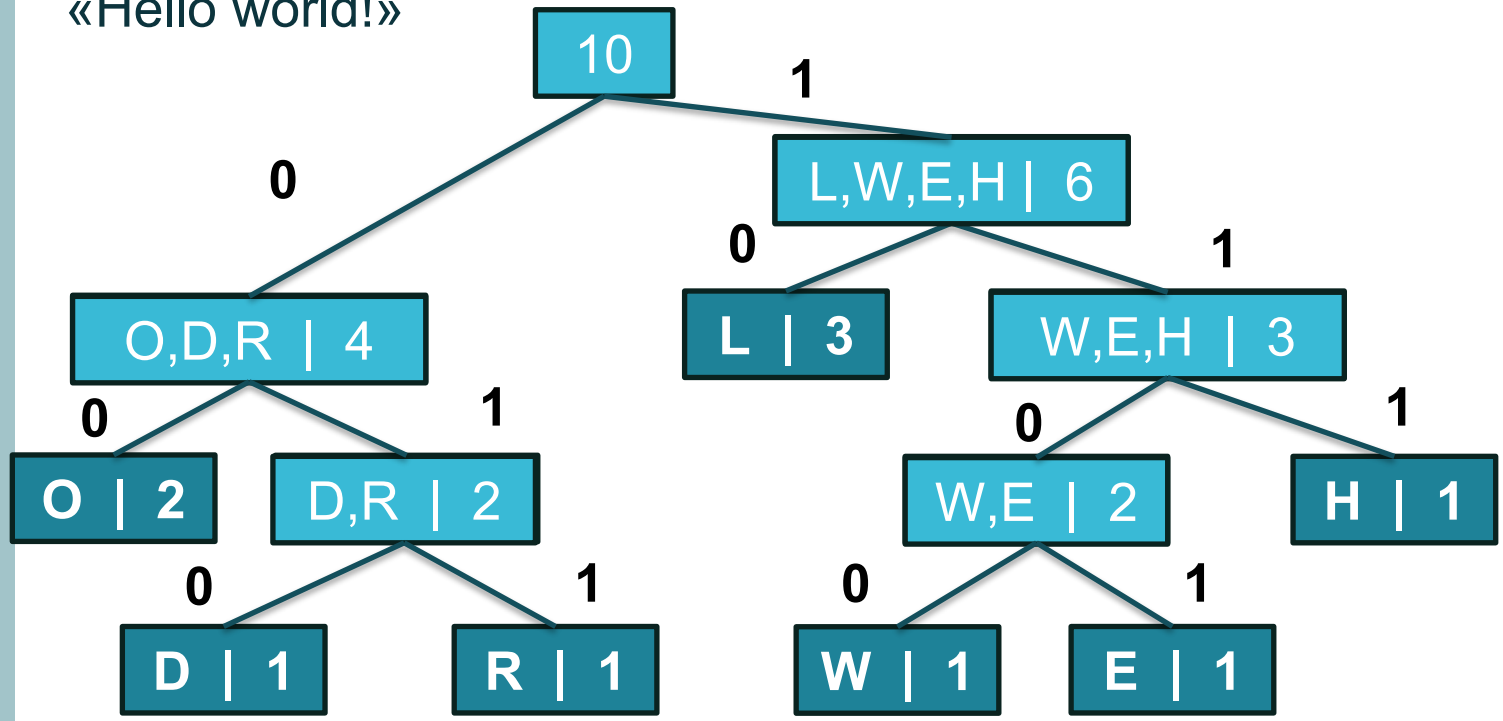
Huffman

«Hello world!»



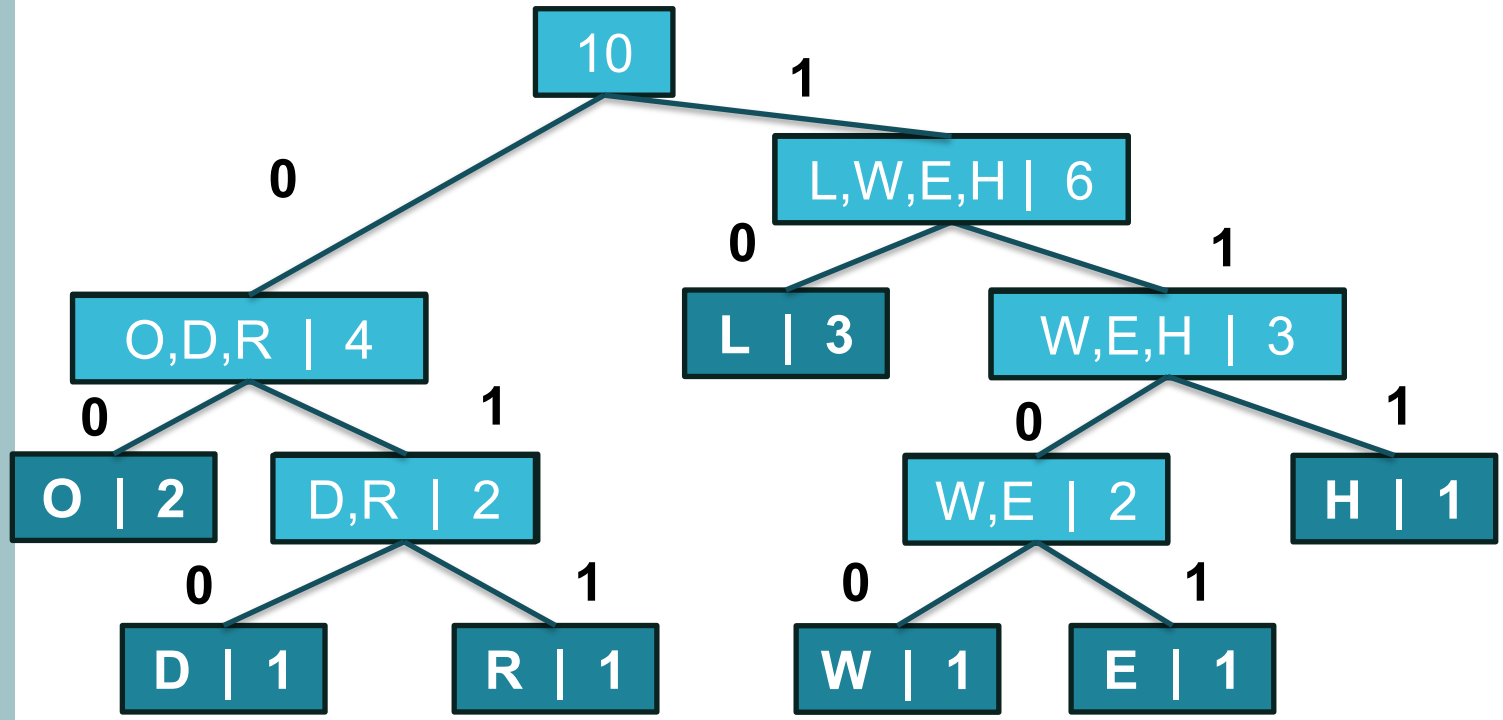
Huffman

«Hello world!»



Huffman

Hva betyr: 111 1101 10 10 00



Ukesoppgaver + oblighjelp