

Análise de Algoritmos

Slides de Paulo Feofiloff

[com erros do coelho e agora também da cris]

Quicksort aleatorizado

CLRS 7.4

Quicksort aleatorizado

PARTICIONE-ALEA(A, p, d)

1 $i \leftarrow \text{RANDOM}(p, d)$

2 $A[i] \leftrightarrow A[d]$

3 **devolva** **PARTICIONE** (A, p, d)

QUICKSORT-ALE (A, p, d)

1 **se** $p < d$

2 **então** $q \leftarrow \text{PARTICIONE-ALEA}(A, p, d)$

3 **QUICKSORT-ALE** ($A, p, q - 1$)

4 **QUICKSORT-ALE** ($A, q + 1, d$)

Análise do consumo medio?

Basta contar o número esperado de comparações na linha 4 do **PARTICIONE**

Particione

Rearranja $A[p..d]$ de modo que $p \leq q \leq d$ e $A[p..q-1] \leq A[q] < A[q+1..d]$

PARTICIONE (A, p, d)

```
1   $x \leftarrow A[d]$       ▷  $x$  é o “pivô”
2   $i \leftarrow p-1$ 
3  para  $j \leftarrow p$  até  $d-1$  faça
4      se  $A[j] \leq x$ 
5          então  $i \leftarrow i + 1$ 
6               $A[i] \leftrightarrow A[j]$ 
7   $A[i+1] \leftrightarrow A[d]$ 
8  devolva  $i + 1$ 
```

Invariantes: no começo de cada iteração de 3–6,

(i0) $A[p..i] \leq x$ (i1) $A[i+1..j-1] > x$ (i2) $A[d] = x$

Consumo de tempo esperado

Suponha $A[p..r]$ permutação de $1..n$.

X_{ab} = número de comparações entre a e b
na linha 4 de **PARTICIONE**

Queremos calcular

$$\begin{aligned} X &= \text{total de comparações "A[j] \leq x"} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n X_{ab} \end{aligned}$$

Consumo de tempo esperado

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \dots, b\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

Consumo de tempo esperado

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \dots, b\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr \{X_{ab}=1\} = \frac{1}{b-a+1} + \frac{1}{b-a+1} = \mathbb{E}[X_{ab}]$$

Consumo de tempo esperado

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \dots, b\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr \{X_{ab}=1\} = \frac{1}{b-a+1} + \frac{1}{b-a+1} = \mathbb{E}[X_{ab}]$$

$$X = \sum_{a=1}^{n-1} \sum_{b=a+1}^n X_{ab}$$

$$\mathbb{E}[X] = \text{????}$$

Consumo de tempo esperado

$$\begin{aligned} E[X] &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n E[X_{ab}] \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \Pr \{X_{ab}=1\} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \frac{2}{b-a+1} \\ &= \sum_{a=1}^{n-1} \sum_{k=1}^{n-a} \frac{2}{k+1} \\ &< \sum_{a=1}^{n-1} 2 \left(\frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{n} \right) \\ &< 2n \left(\frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{n} \right) < 2n (1 + \ln n) \end{aligned}$$

CLRS (A.7), p.1060

Conclusões

O consumo de tempo esperado do algoritmo
QUICKSORT-ALE é $O(n \log n)$.

Do **exercício 7.4-4 do CLRS** temos que

O consumo de tempo esperado do algoritmo
QUICKSORT-ALE é $\Theta(n \log n)$.

Heap

Um vetor $A[1 \dots m]$ é um (max-)heap se

$$A[\lfloor i/2 \rfloor] \geq A[i]$$

para todo $i = 2, 3, \dots, m$.

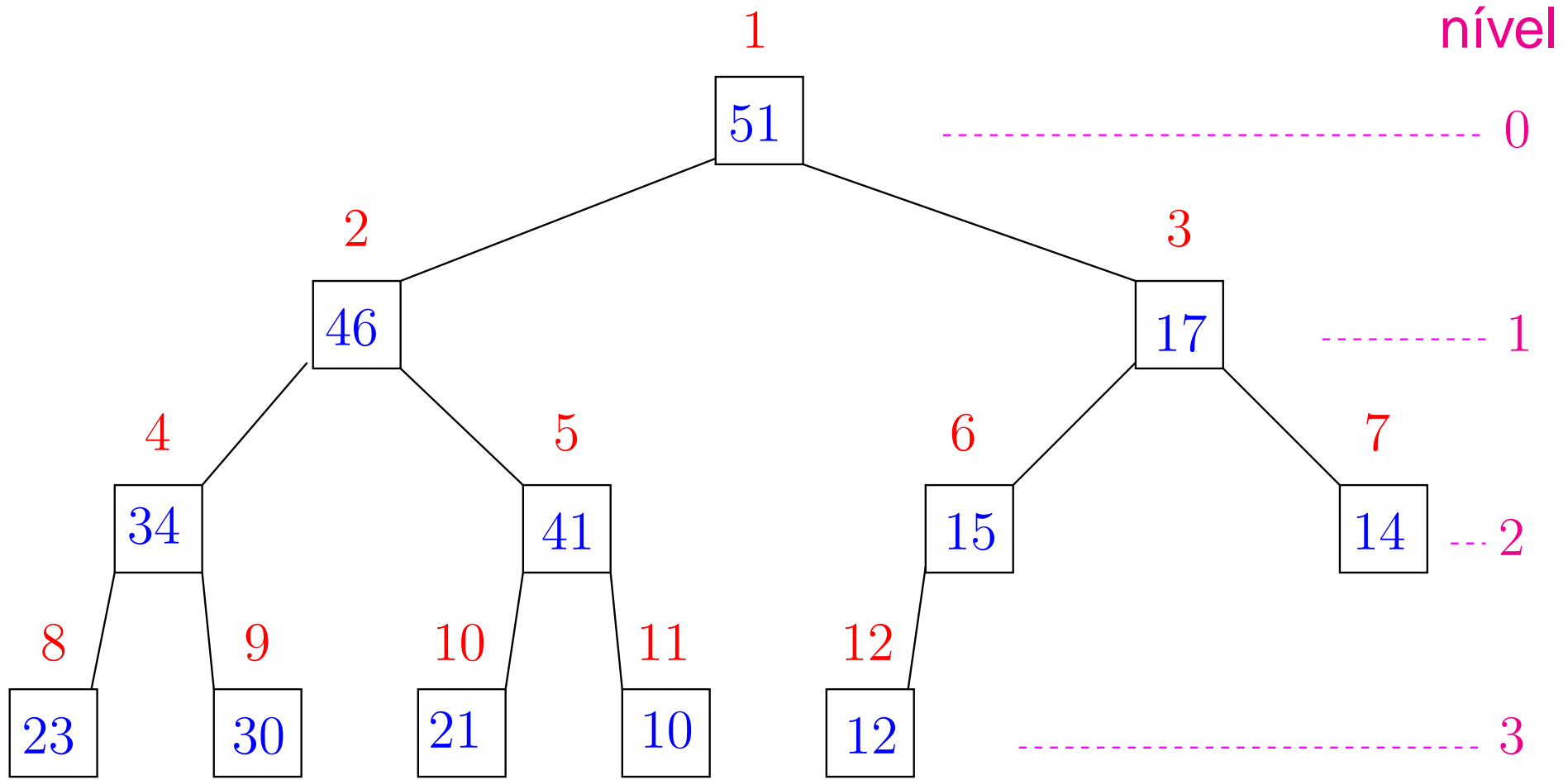
De uma forma mais geral, $A[j \dots m]$ é um heap se

$$A[\lfloor i/2 \rfloor] \geq A[i]$$

para todo $i = 2j, 2j + 1, 4j, \dots, 4j + 3, 8j, \dots, 8j + 7, \dots$

Neste caso também diremos que a subárvore com raiz j é um heap.

Exemplo



1	2	3	4	5	6	7	8	9	10	11	12
51	46	17	34	41	15	14	23	30	21	10	12

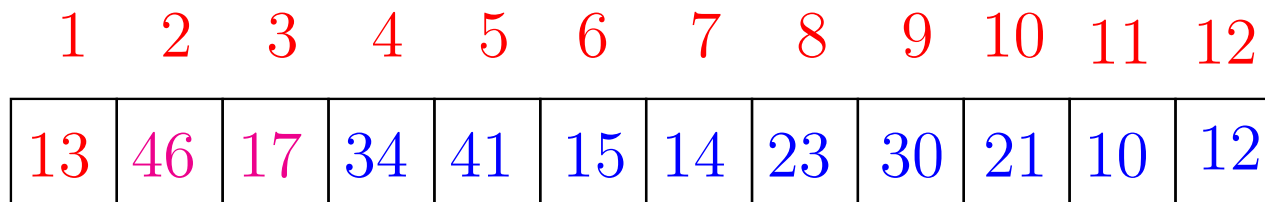
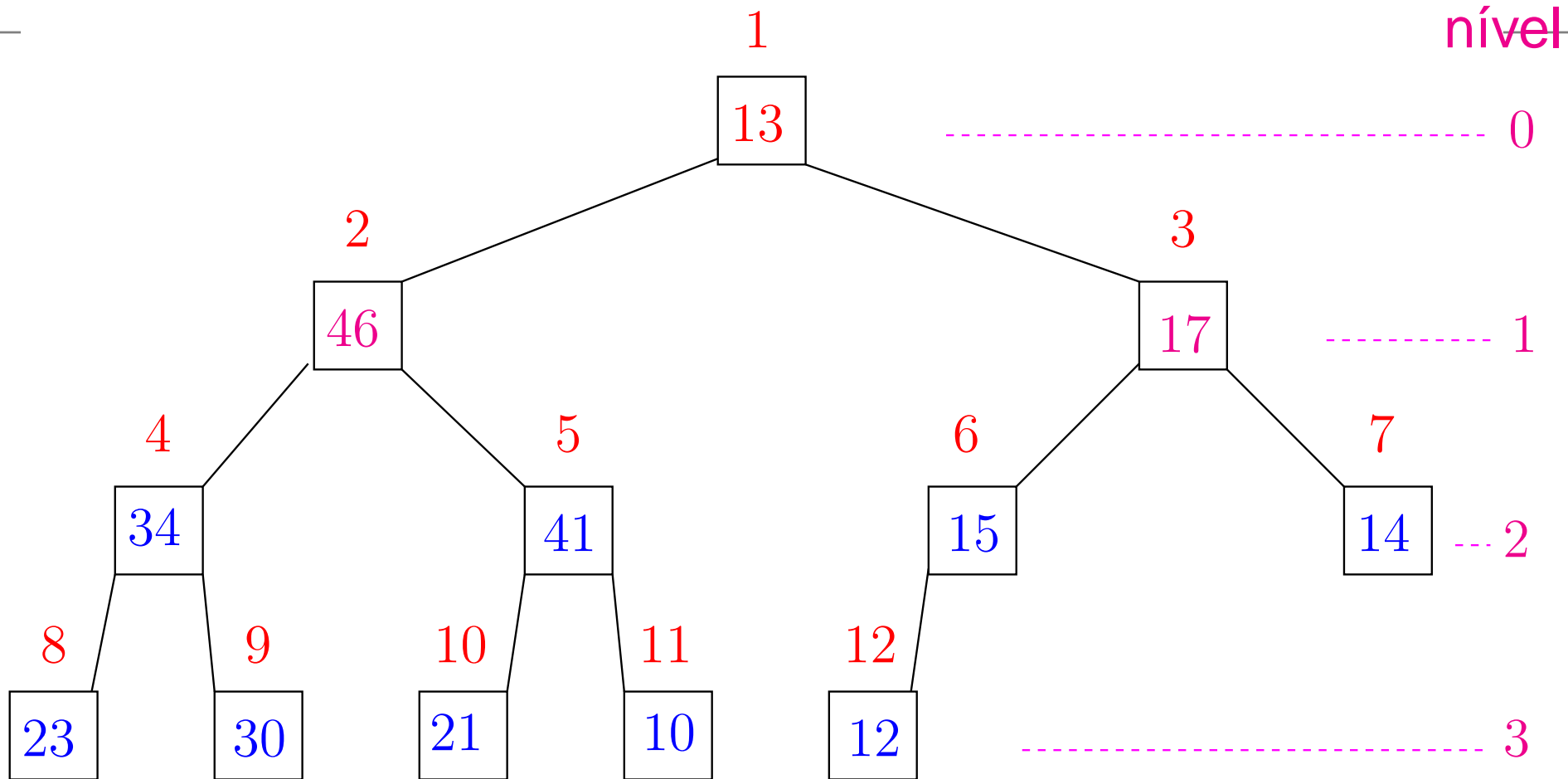
Desce-Heap

Recebe $A[1..m]$ e $i \geq 1$ tais que subárvores com raiz $2i$ e $2i + 1$ são heaps e **rearranja** A de modo que subárvore com raiz i seja heap.

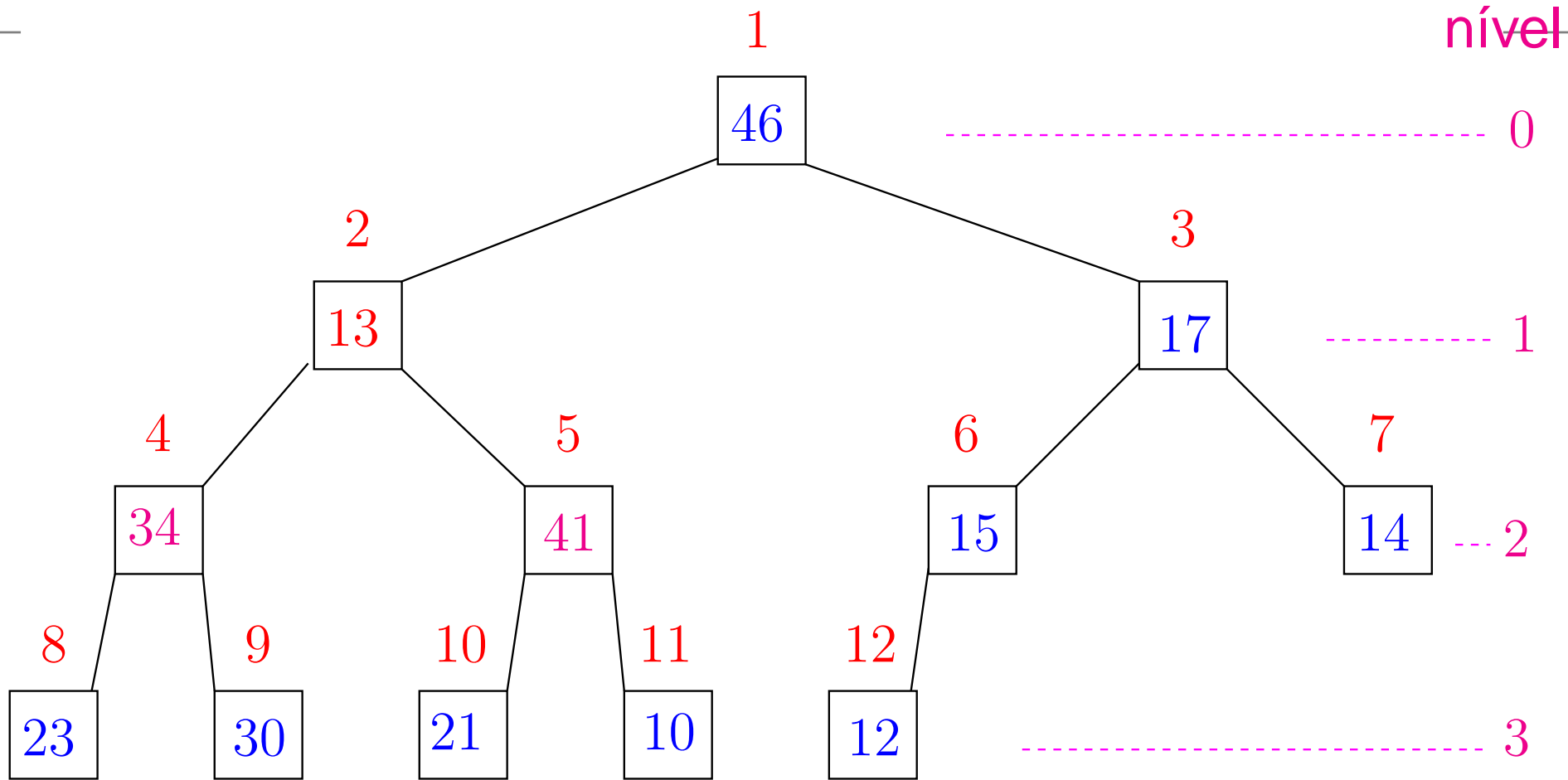
DESCE-HEAP (A, m, i)

```
1   $e \leftarrow 2i$ 
2   $d \leftarrow 2i + 1$ 
3  se  $e \leq m$  e  $A[e] > A[i]$ 
4      então  $maior \leftarrow e$ 
5      senão  $maior \leftarrow i$ 
6  se  $d \leq m$  e  $A[d] > A[maior]$ 
7      então  $maior \leftarrow d$ 
8  se  $maior \neq i$ 
9      então  $A[i] \leftrightarrow A[maior]$ 
10     DESCE-HEAP ( $A, m, maior$ )
```

Simulação



Simulação



1	2	3	4	5	6	7	8	9	10	11	12
46	13	17	34	41	15	14	23	30	21	10	12

nível

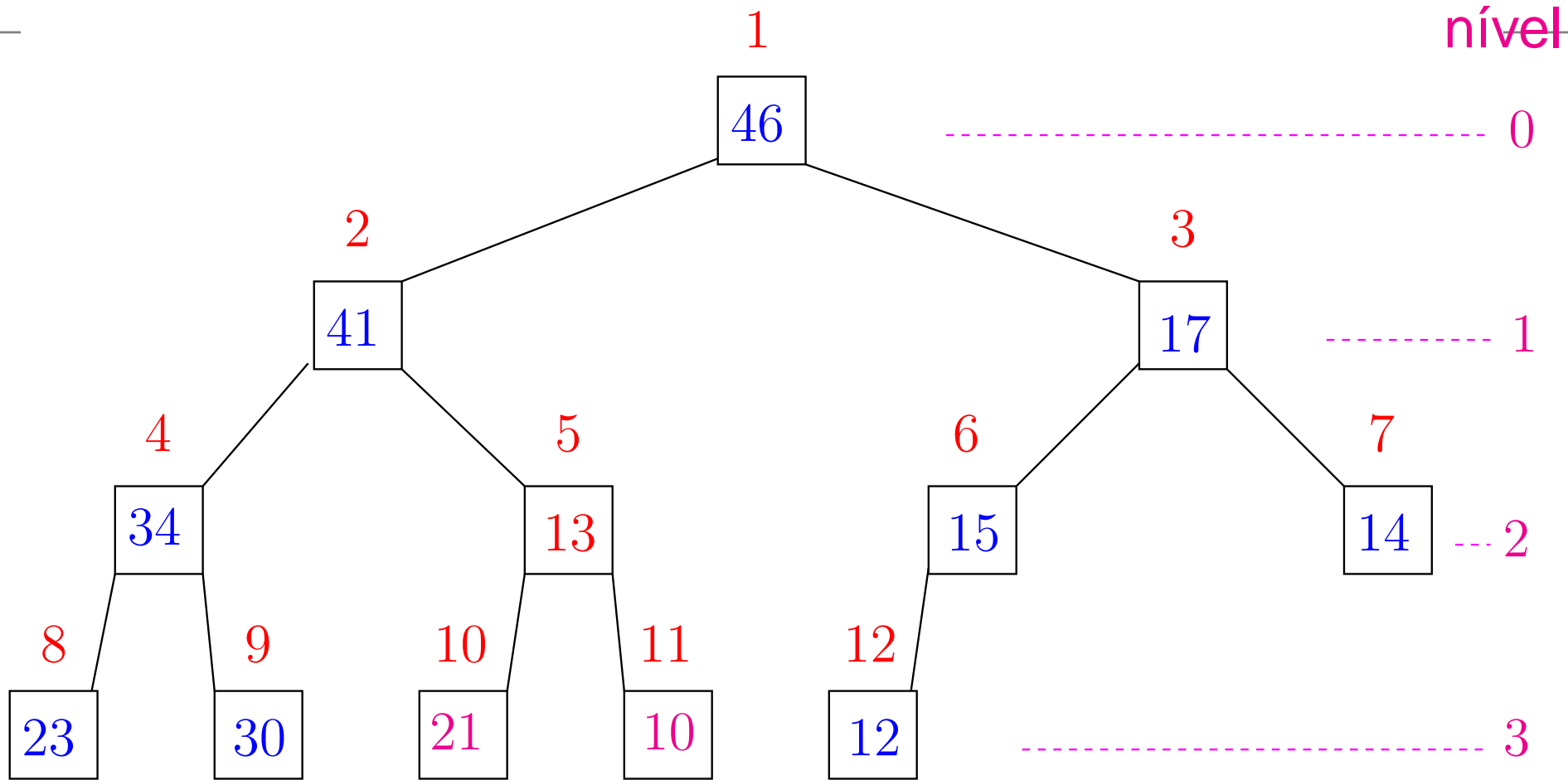
0

1

2

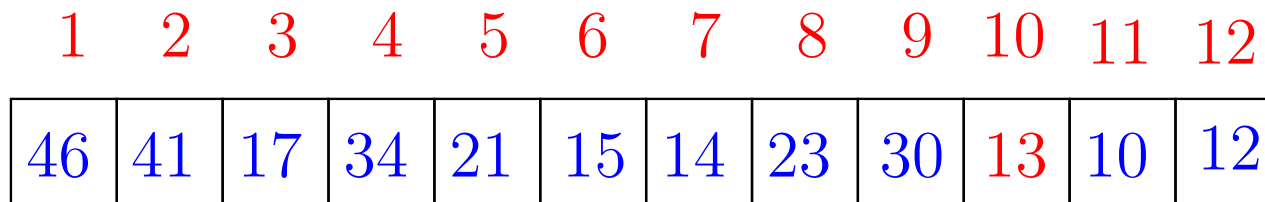
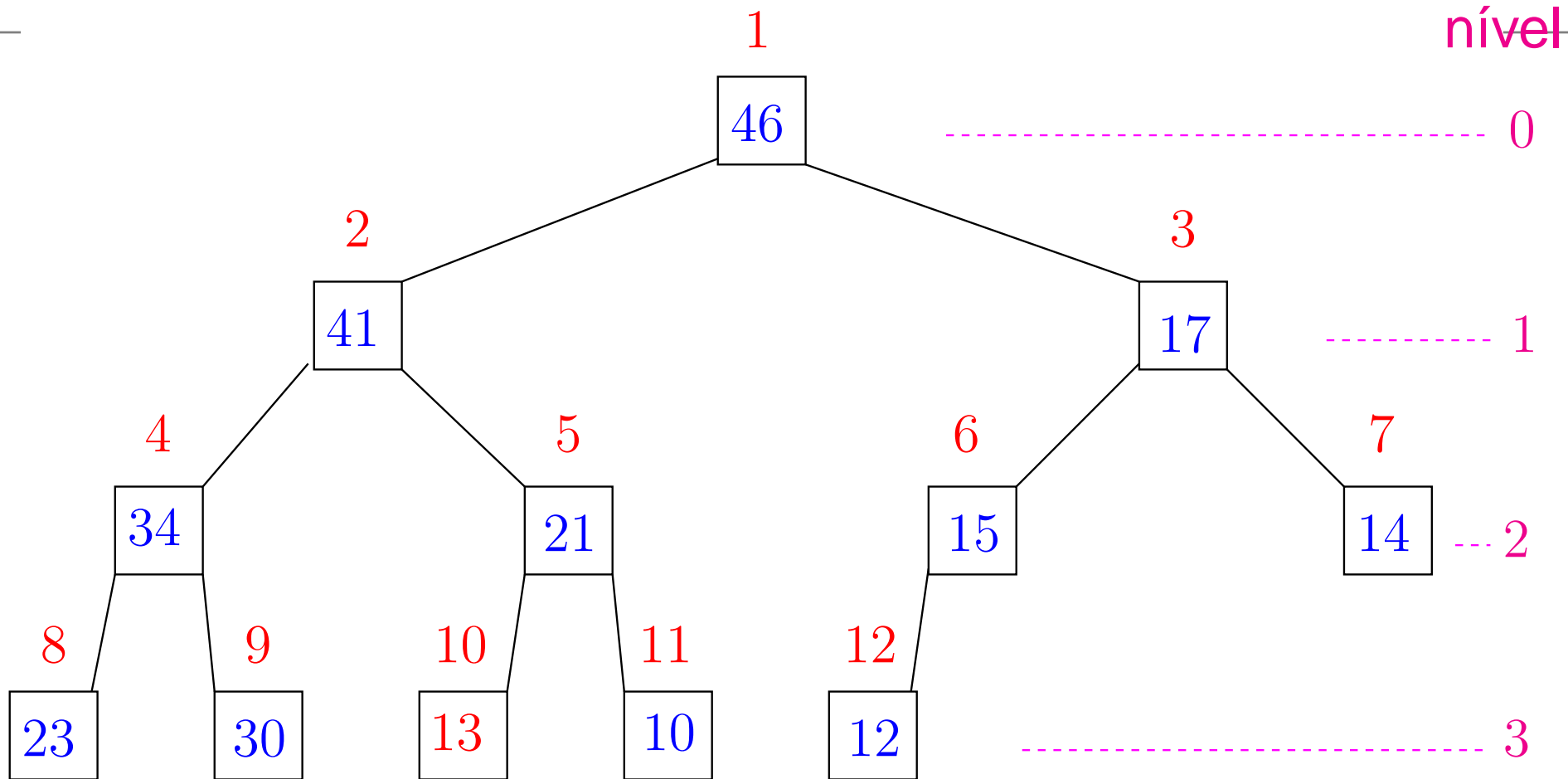
3

Simulação

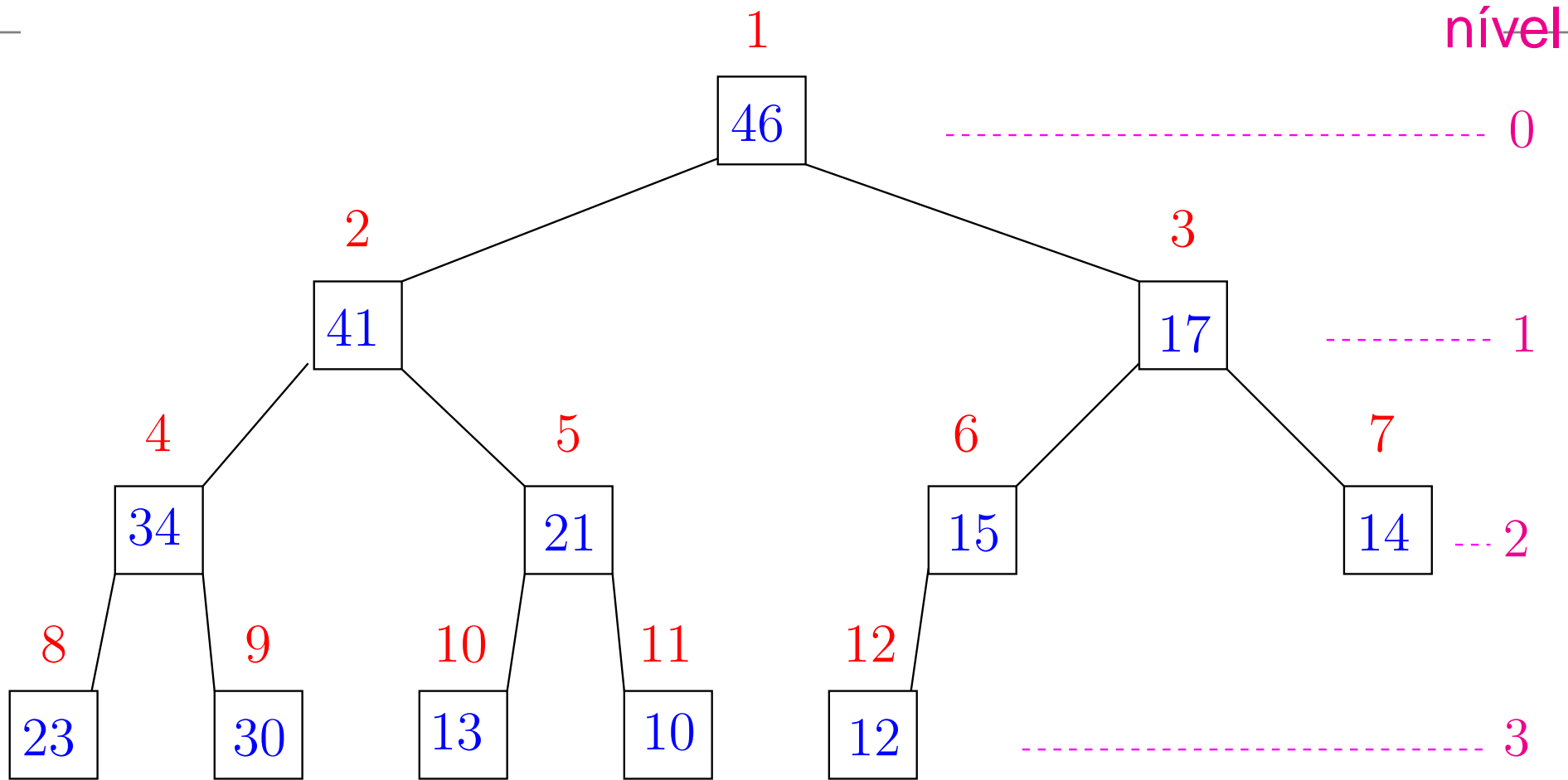


1	2	3	4	5	6	7	8	9	10	11	12
46	41	17	34	13	15	14	23	30	21	10	12

Simulação



Simulação



1	2	3	4	5	6	7	8	9	10	11	12
46	41	17	34	21	15	14	23	30	13	10	12

Consumo de tempo

$h :=$ altura de $i = \lfloor \lg \frac{m+1}{i+1} \rfloor$

$T(h) :=$ consumo de tempo no pior caso

linha	todas as execuções da linha
1-3	$= 3 \Theta(1)$
4-5	$= 2 O(1)$
6	$= \Theta(1)$
7	$= O(1)$
8	$= \Theta(1)$
9	$= O(1)$
10	$\leq T(h - 1)$
total	$\leq T(h - 1) + \Theta(5) + O(2)$

Consumo de tempo

$h :=$ altura de $i = \lfloor \lg \frac{m+1}{i+1} \rfloor$

$T(h) :=$ consumo de tempo no pior caso

Recorrência associada:

$$T(h) \leq T(h - 1) + \Theta(1),$$

pois altura de *maior* é $h - 1$.

Consumo de tempo

$h :=$ altura de $i = \lfloor \lg \frac{m+1}{i+1} \rfloor$

$T(h) :=$ consumo de tempo no pior caso

Recorrência associada:

$$T(h) \leq T(h - 1) + \Theta(1),$$

pois altura de *maior* é $h - 1$.

Solução assintótica: $T(n)$ é ???.

Consumo de tempo

$h :=$ altura de $i = \lfloor \lg \frac{m+1}{i+1} \rfloor$

$T(h) :=$ consumo de tempo no pior caso

Recorrência associada:

$$T(h) \leq T(h - 1) + \Theta(1),$$

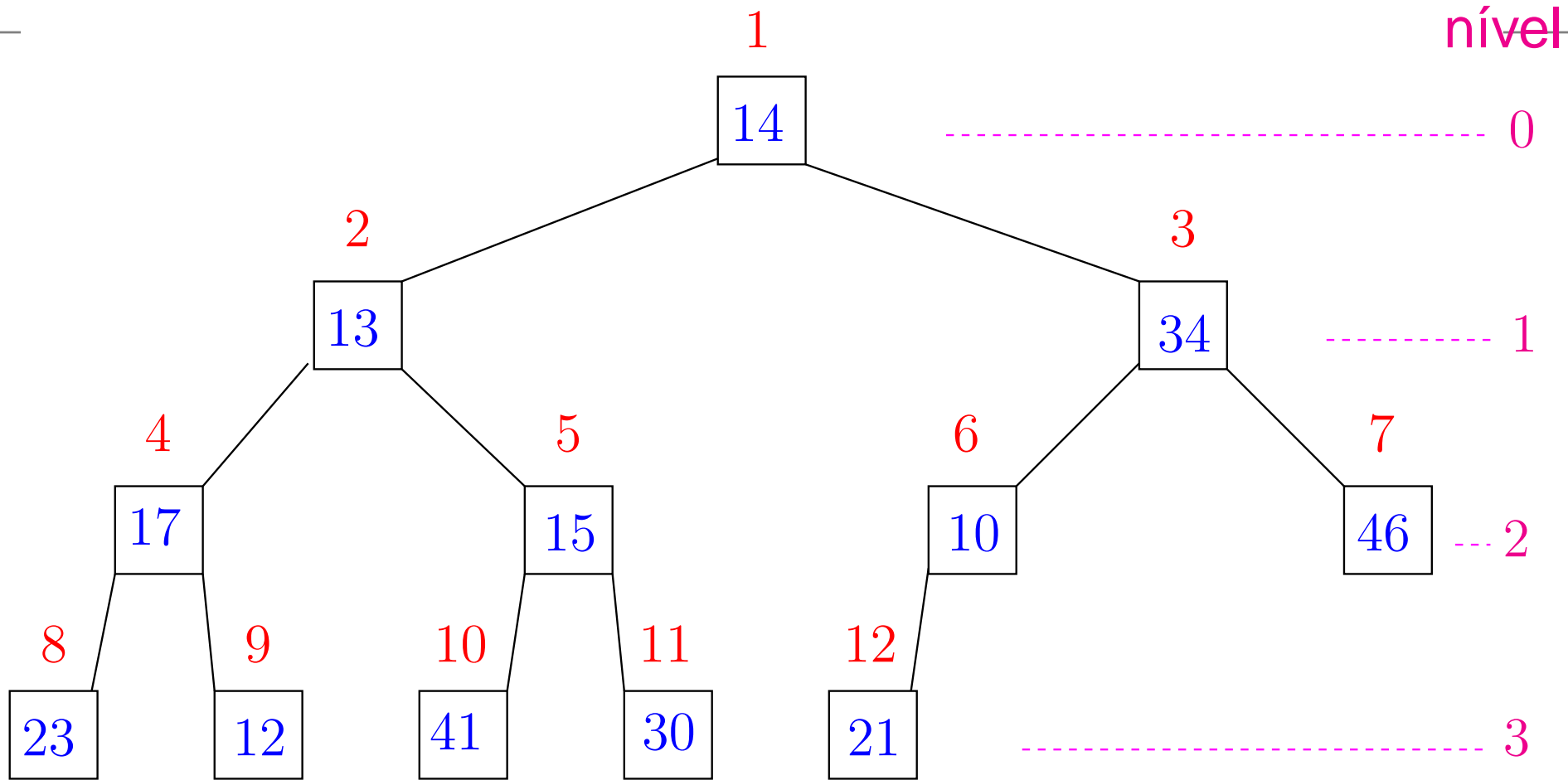
pois altura de *maior* é $h - 1$.

Solução assintótica: $T(n)$ é $O(h)$.

Como $h \leq \lg m$, podemos dizer que:

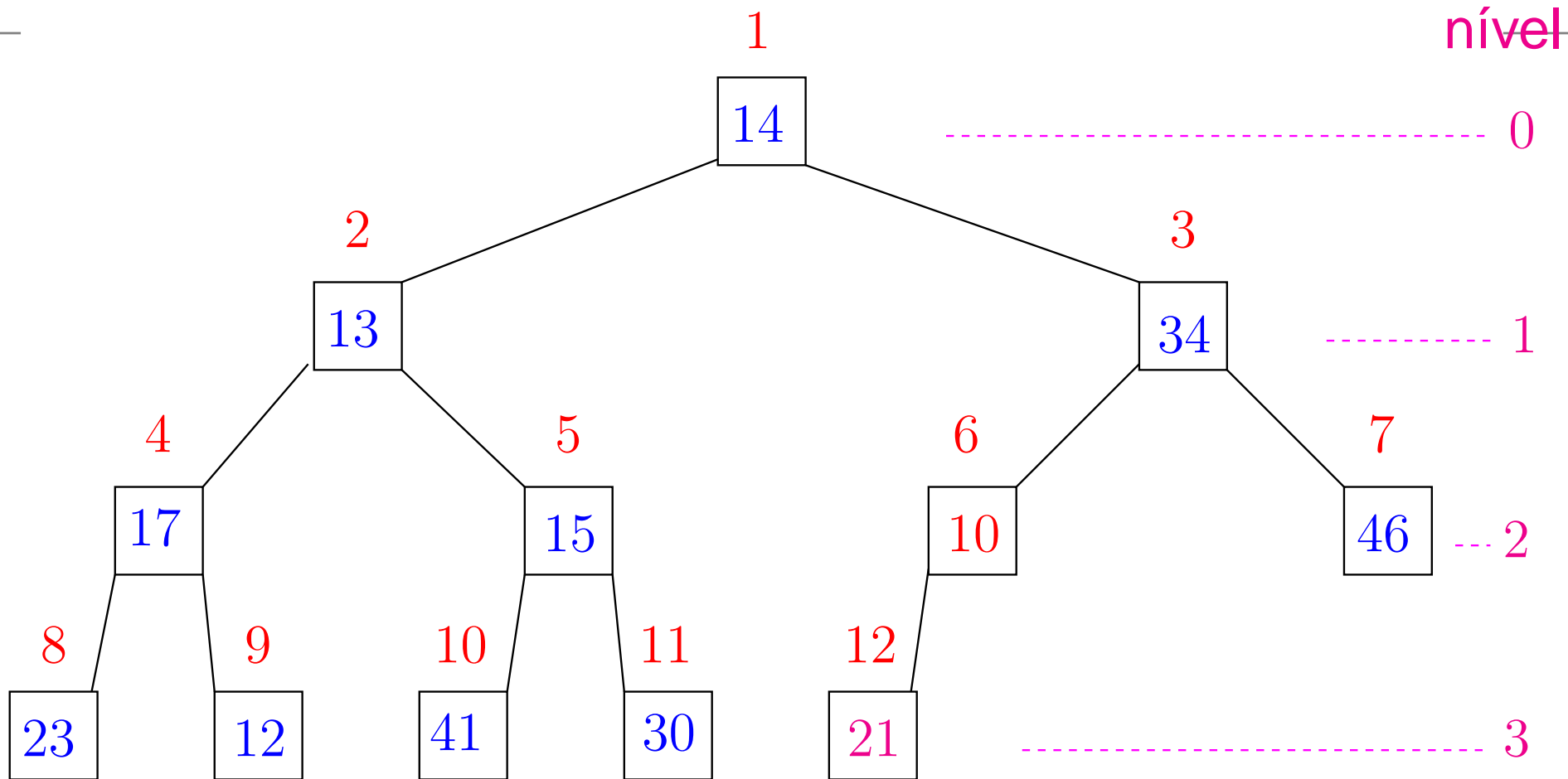
O consumo de tempo do algoritmo **DESCE-HEAP** é $O(\lg m)$ (ou melhor ainda, $O(\lg \frac{m+1}{i+1})$).

Construção de um heap



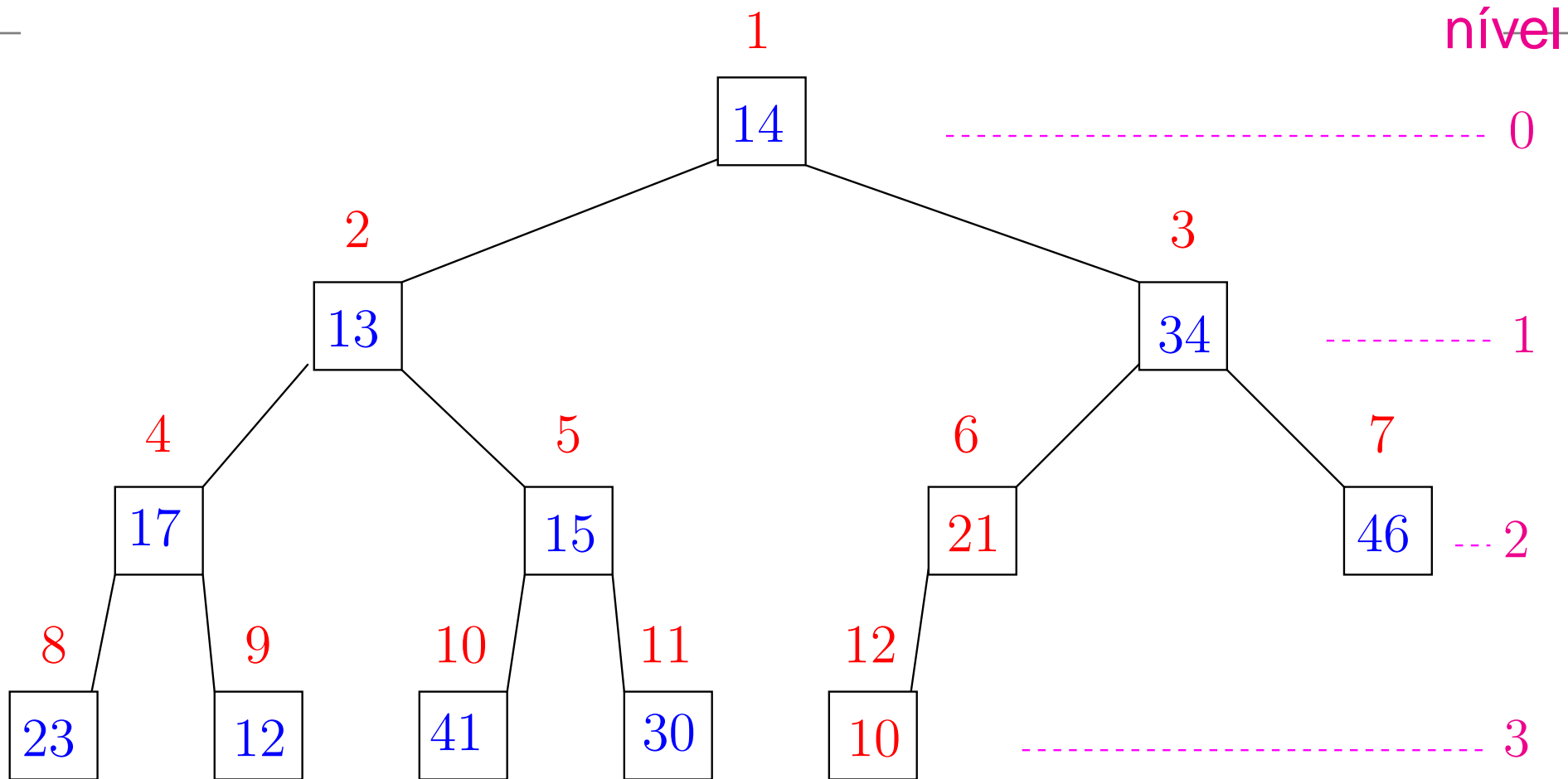
1	2	3	4	5	6	7	8	9	10	11	12
14	13	34	17	15	10	46	23	12	41	30	21

Construção de um heap



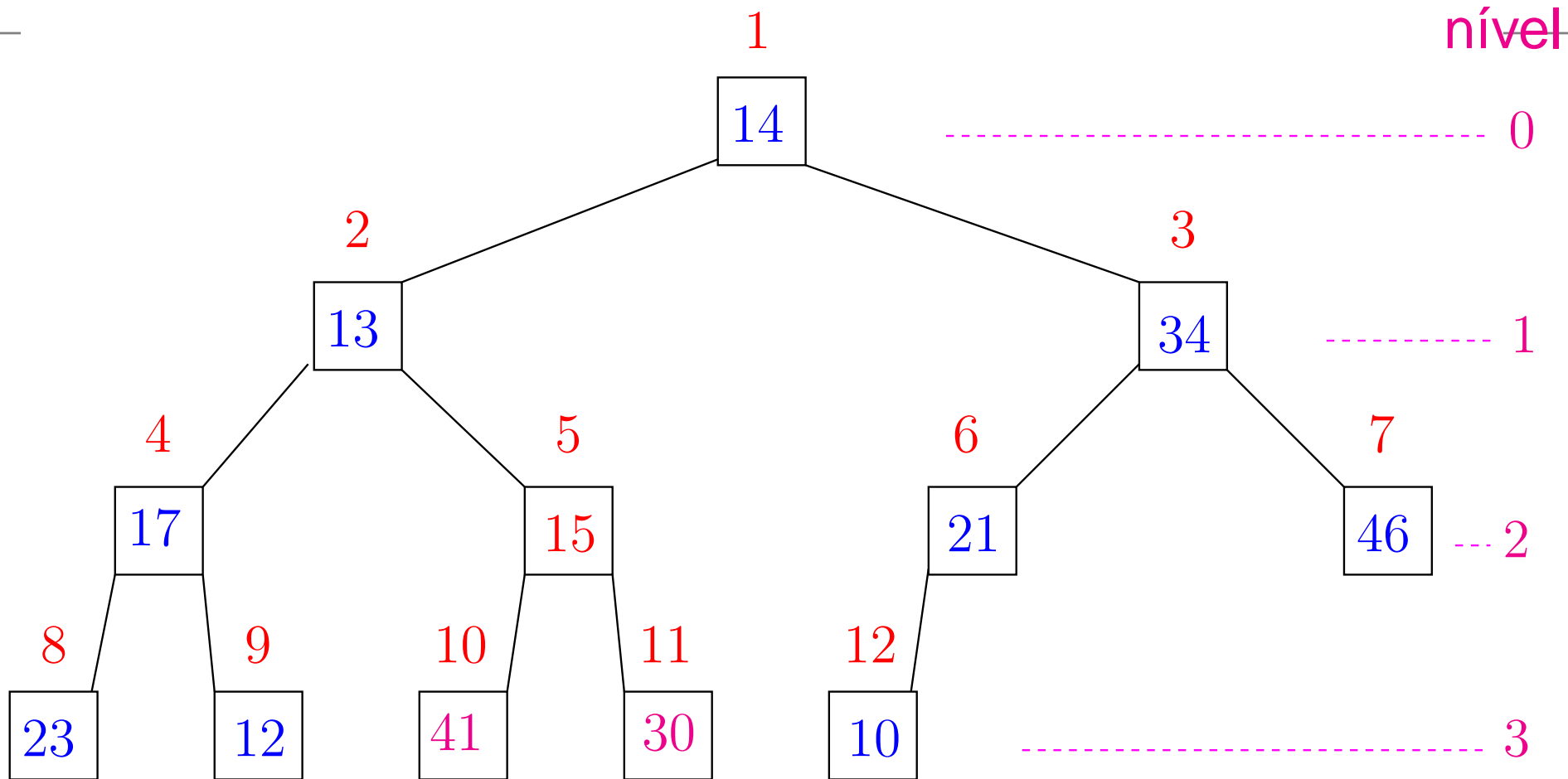
1	2	3	4	5	6	7	8	9	10	11	12
14	13	34	17	15	10	46	23	12	41	30	21

Construção de um heap



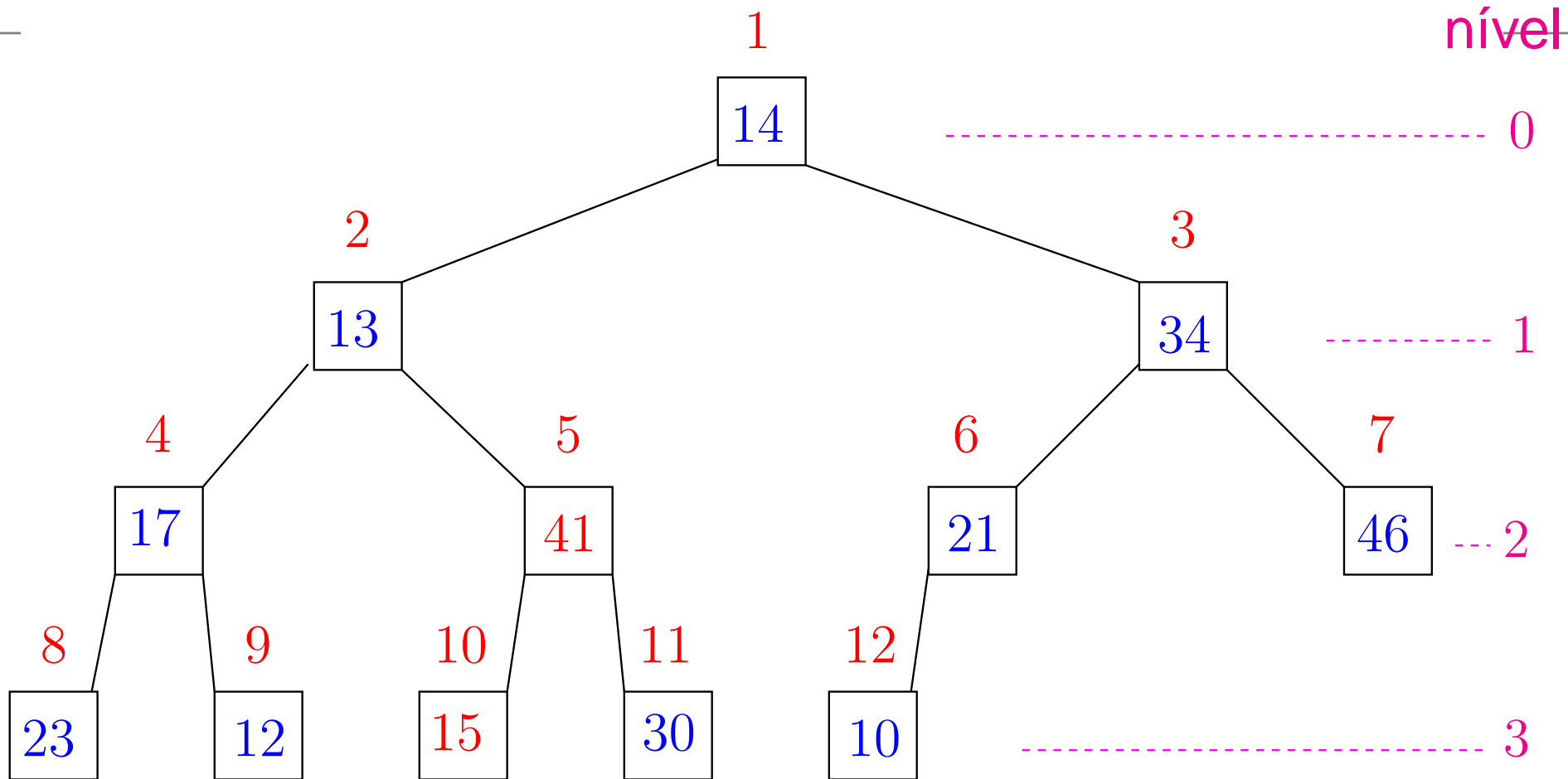
1	2	3	4	5	6	7	8	9	10	11	12
14	13	34	17	15	21	46	23	12	41	30	10

Construção de um heap



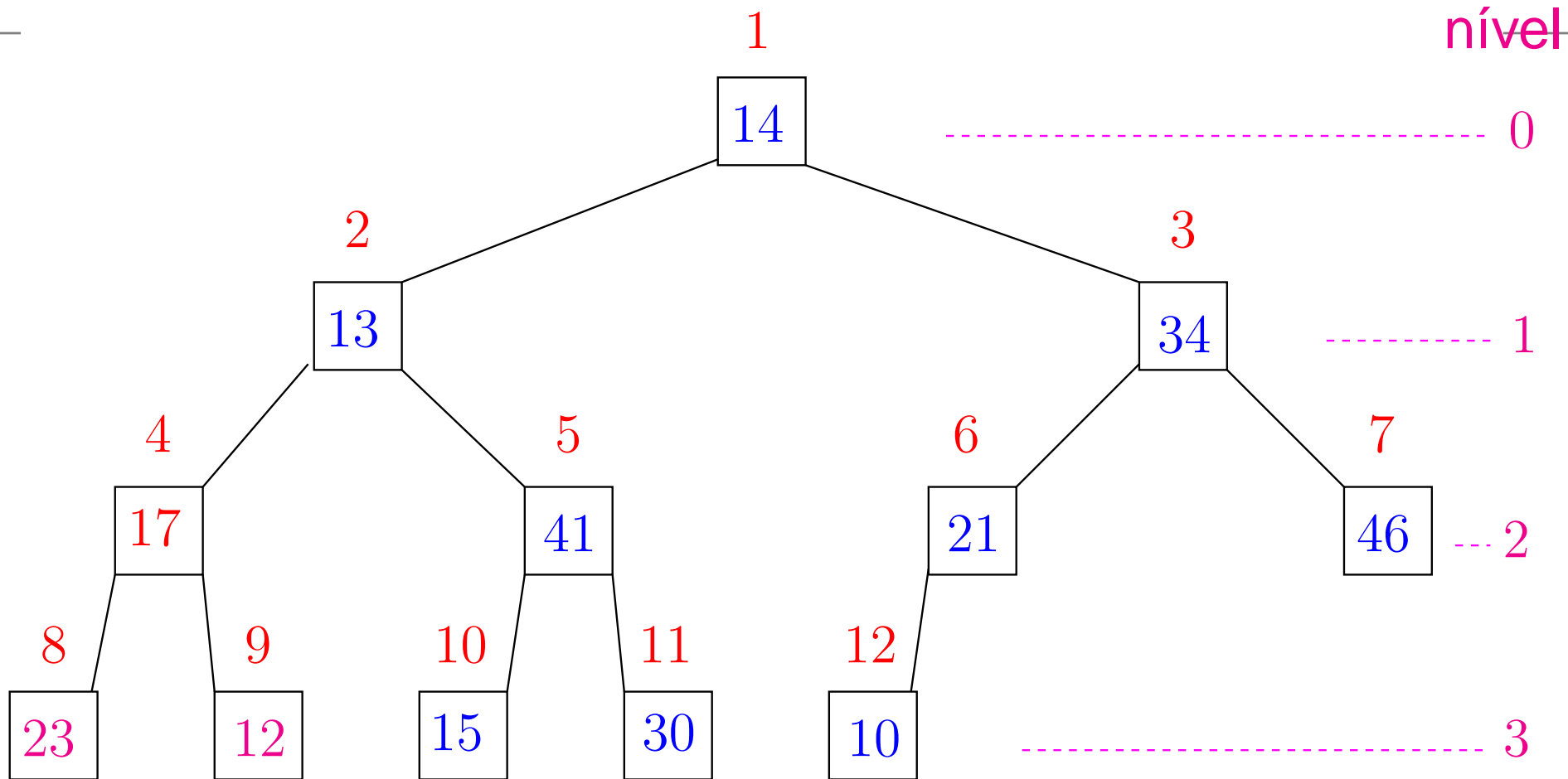
1	2	3	4	5	6	7	8	9	10	11	12
14	13	34	17	15	21	46	23	12	41	30	10

Construção de um heap



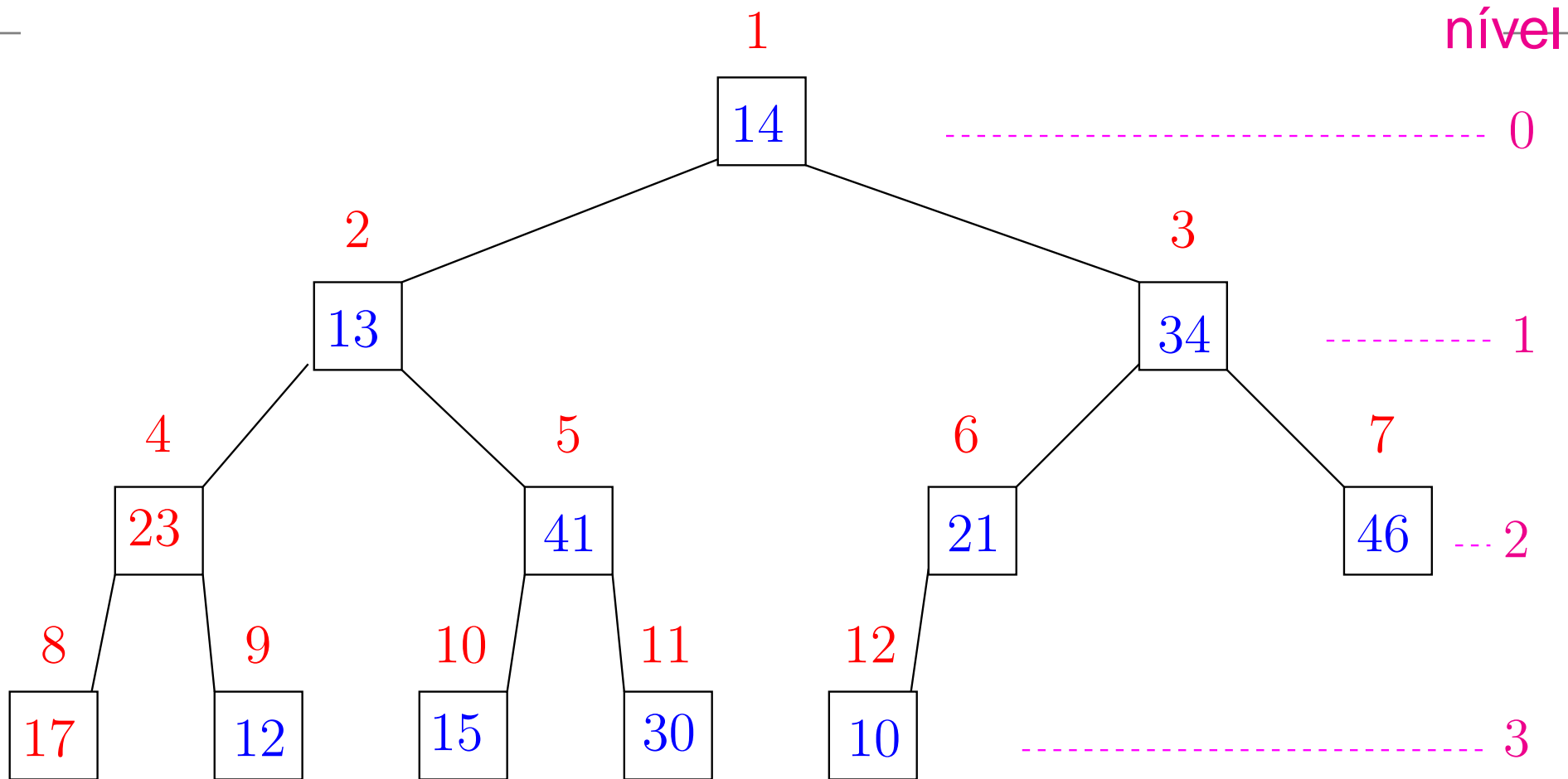
1	2	3	4	5	6	7	8	9	10	11	12
14	13	34	17	41	21	46	23	12	15	30	10

Construção de um heap



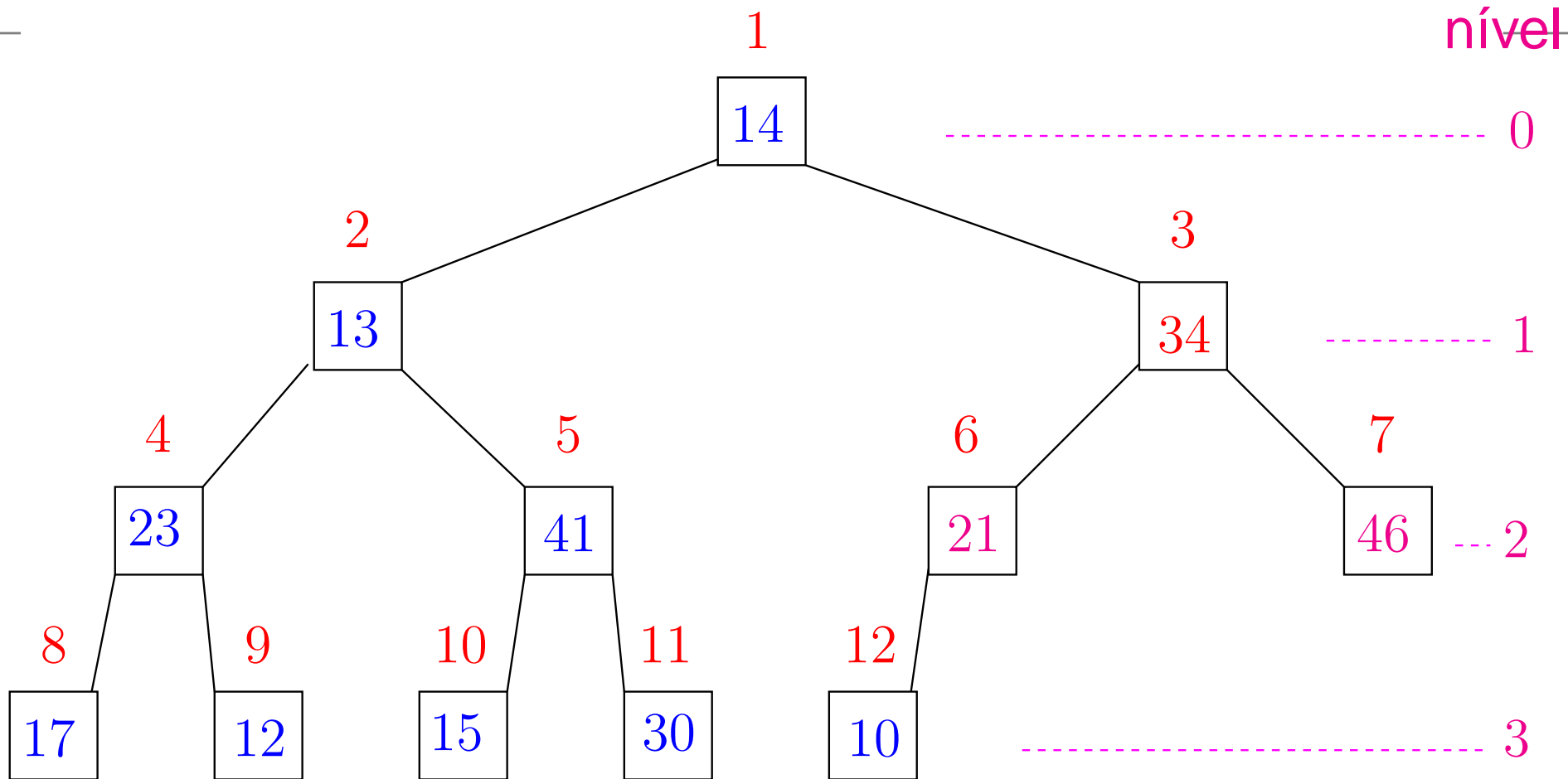
1	2	3	4	5	6	7	8	9	10	11	12
14	13	34	17	41	21	46	23	12	15	30	10

Construção de um heap



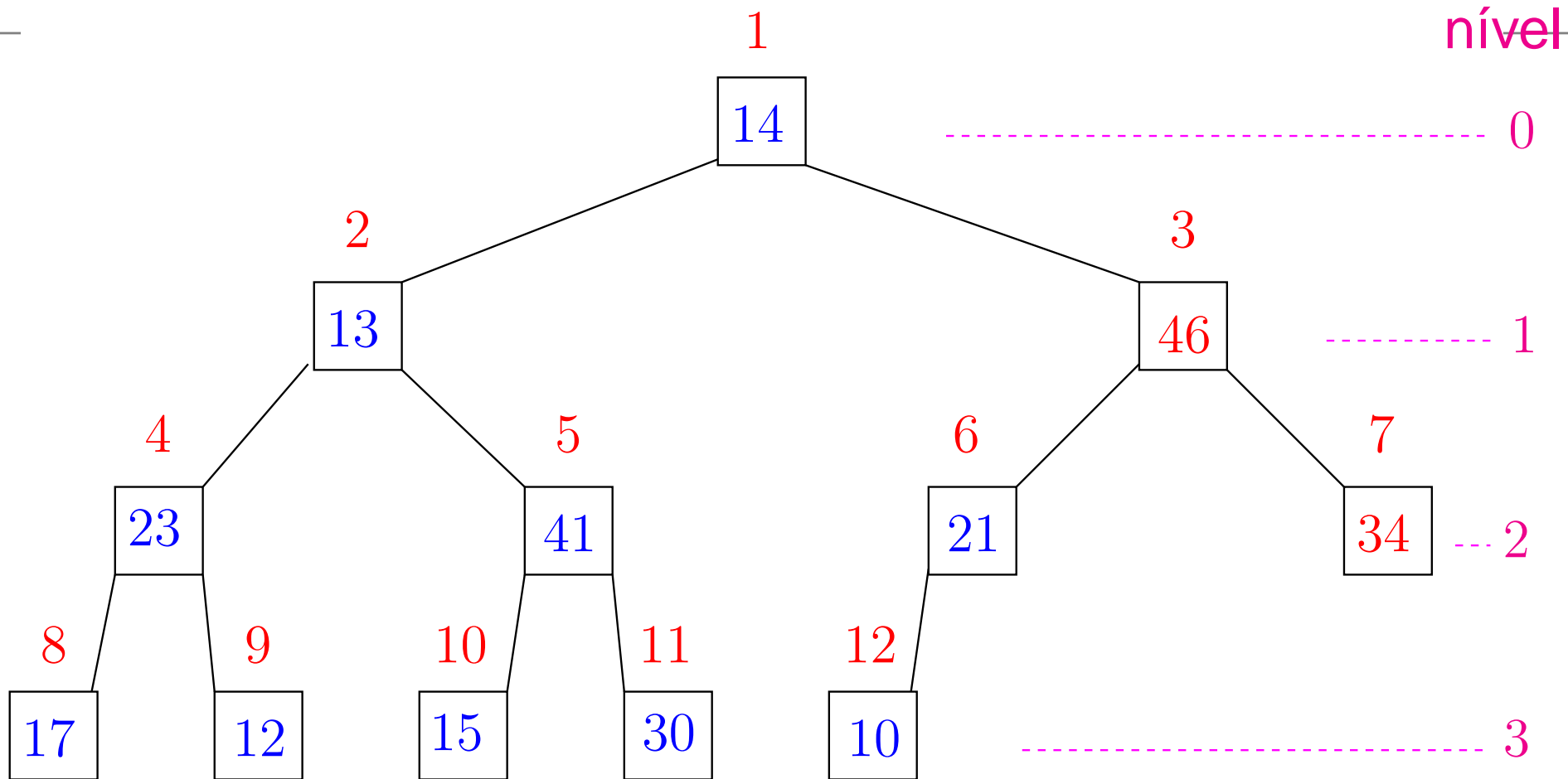
1	2	3	4	5	6	7	8	9	10	11	12
14	13	34	23	41	21	46	17	12	15	30	10

Construção de um heap



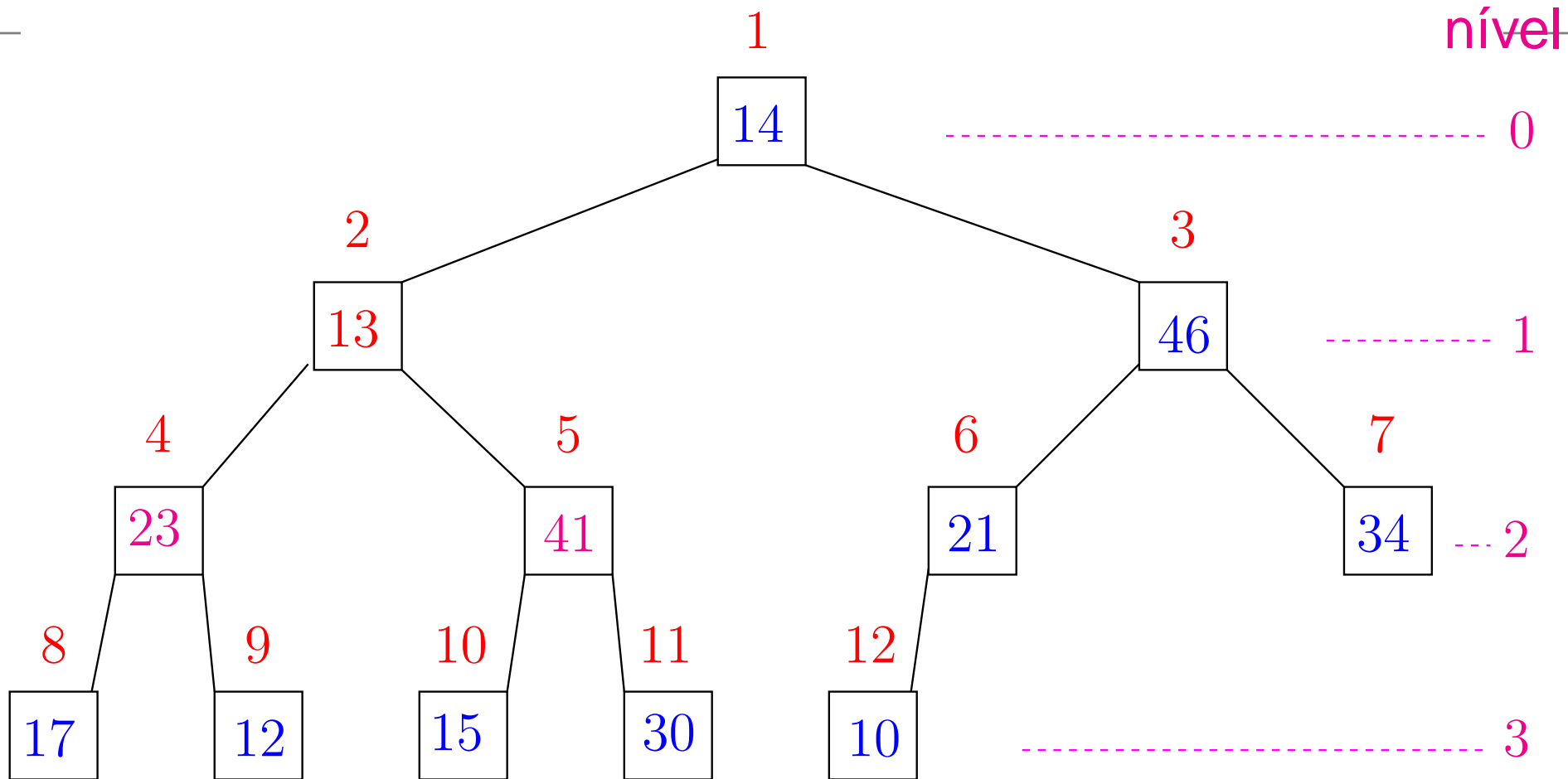
1	2	3	4	5	6	7	8	9	10	11	12
14	13	34	23	41	21	46	17	12	15	30	10

Construção de um heap



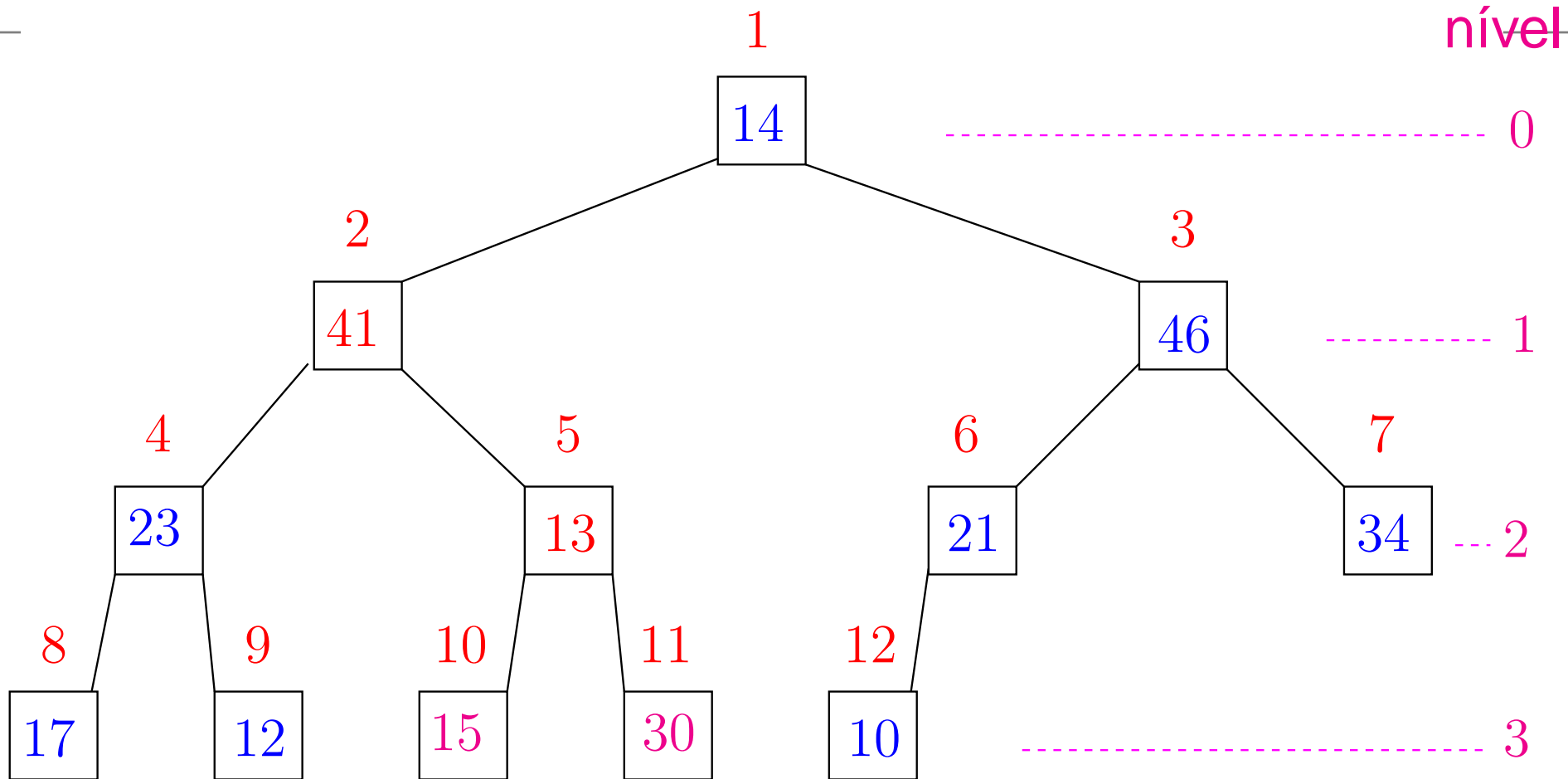
1	2	3	4	5	6	7	8	9	10	11	12
14	13	46	23	41	21	34	17	12	15	30	10

Construção de um heap



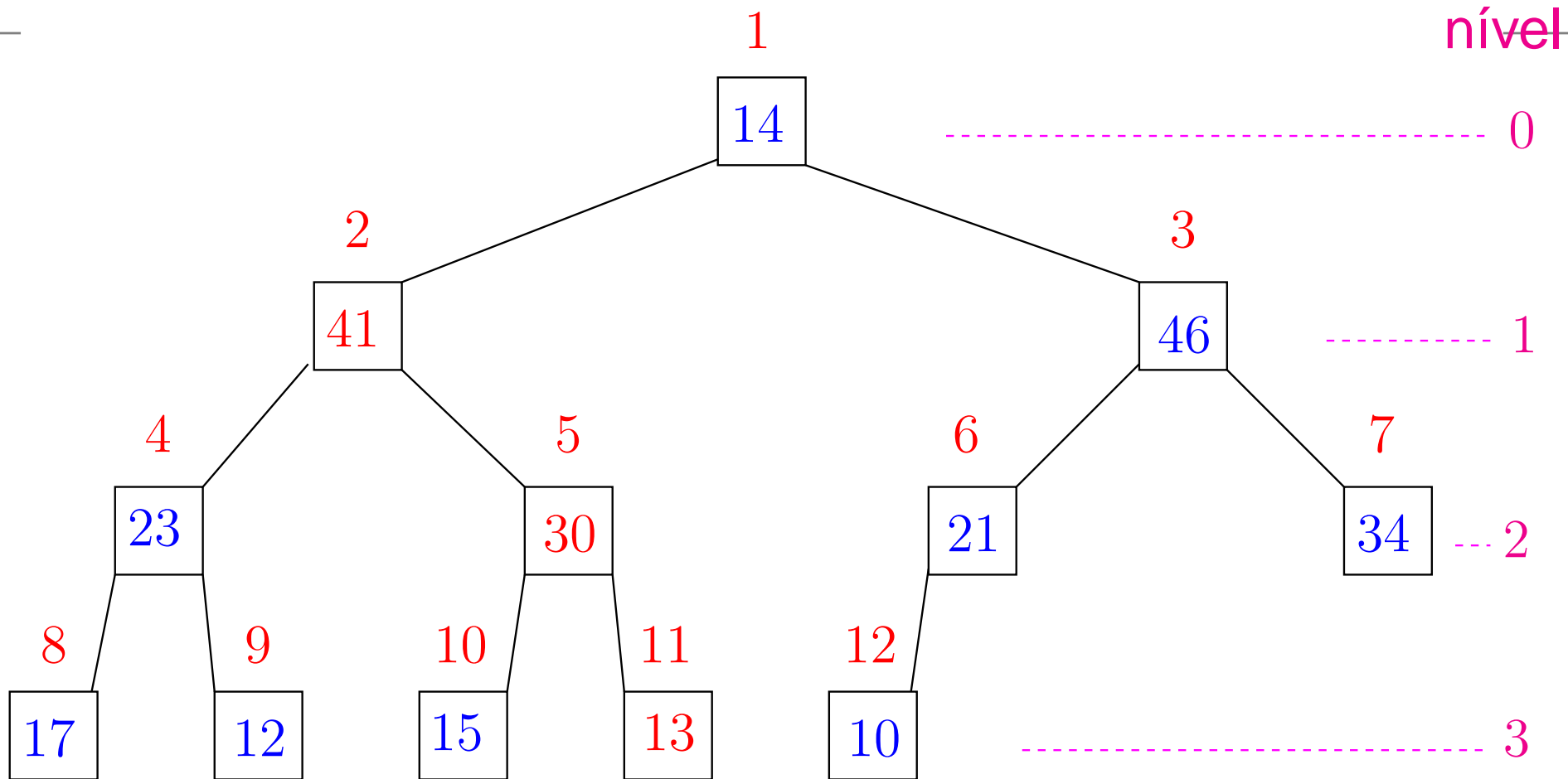
1	2	3	4	5	6	7	8	9	10	11	12
14	13	46	23	41	21	34	17	12	15	30	10

Construção de um heap



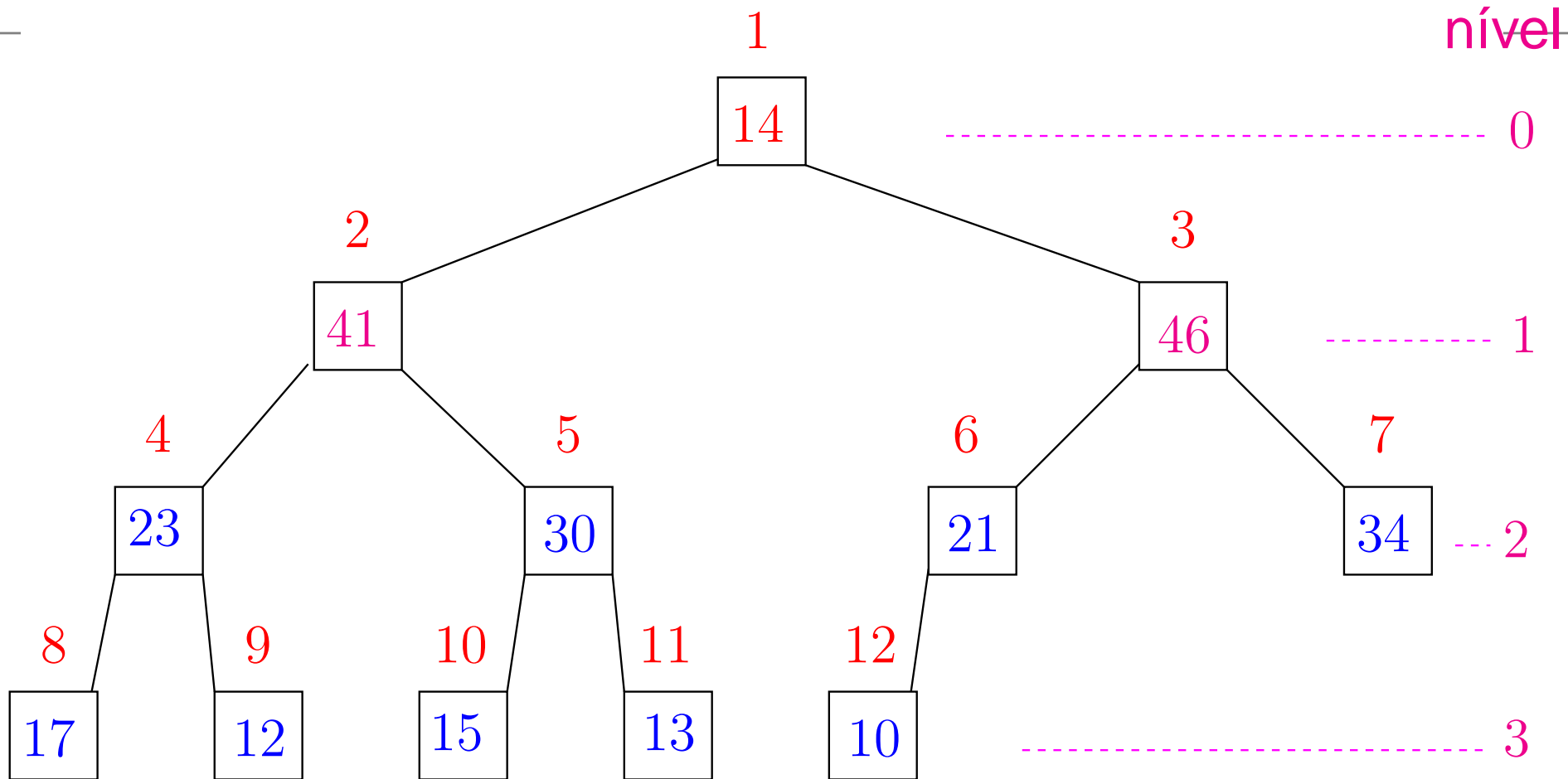
1	2	3	4	5	6	7	8	9	10	11	12
14	41	46	23	13	21	34	17	12	15	30	10

Construção de um heap



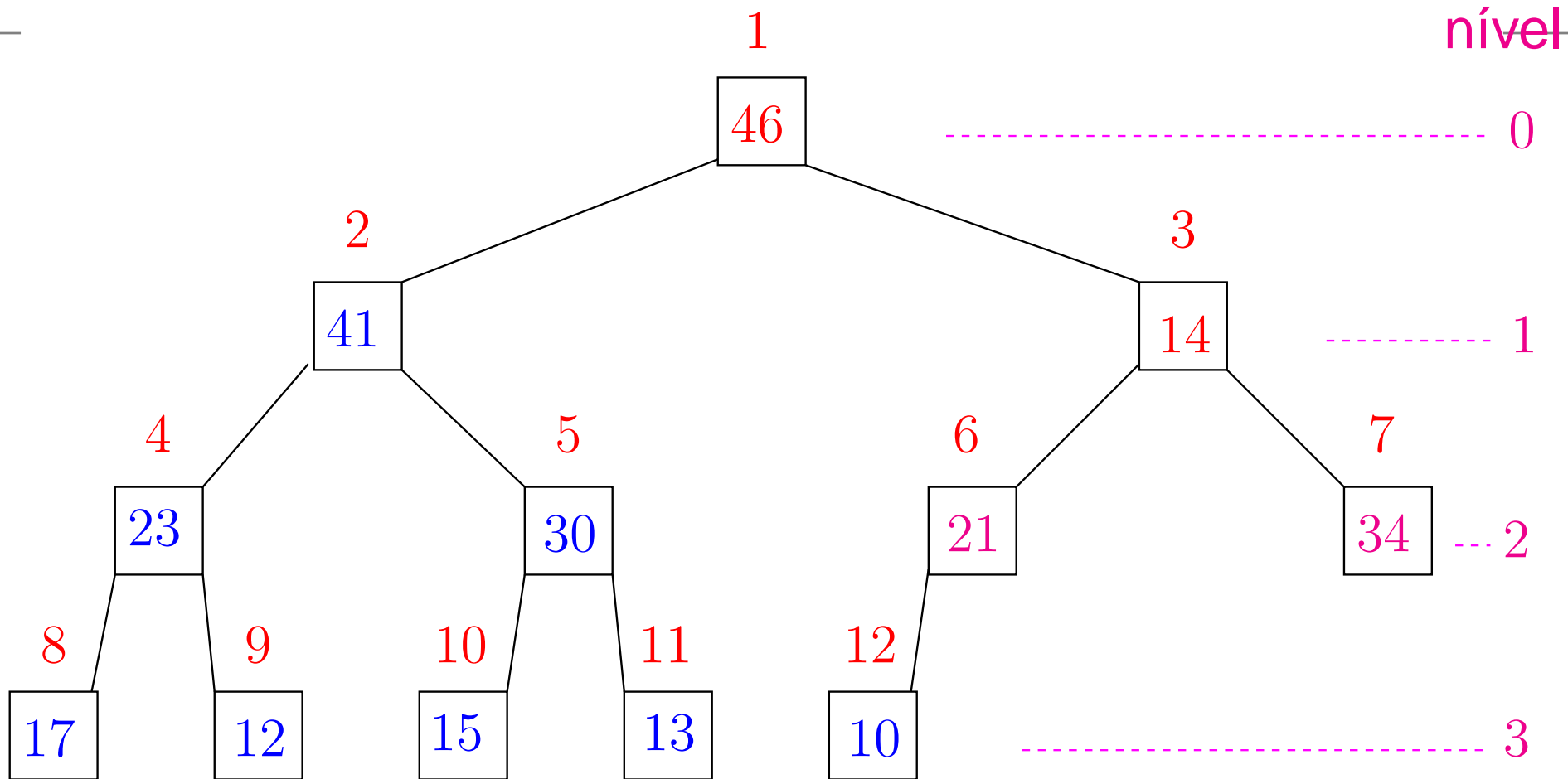
1	2	3	4	5	6	7	8	9	10	11	12
14	41	46	23	30	21	34	17	12	15	13	10

Construção de um heap



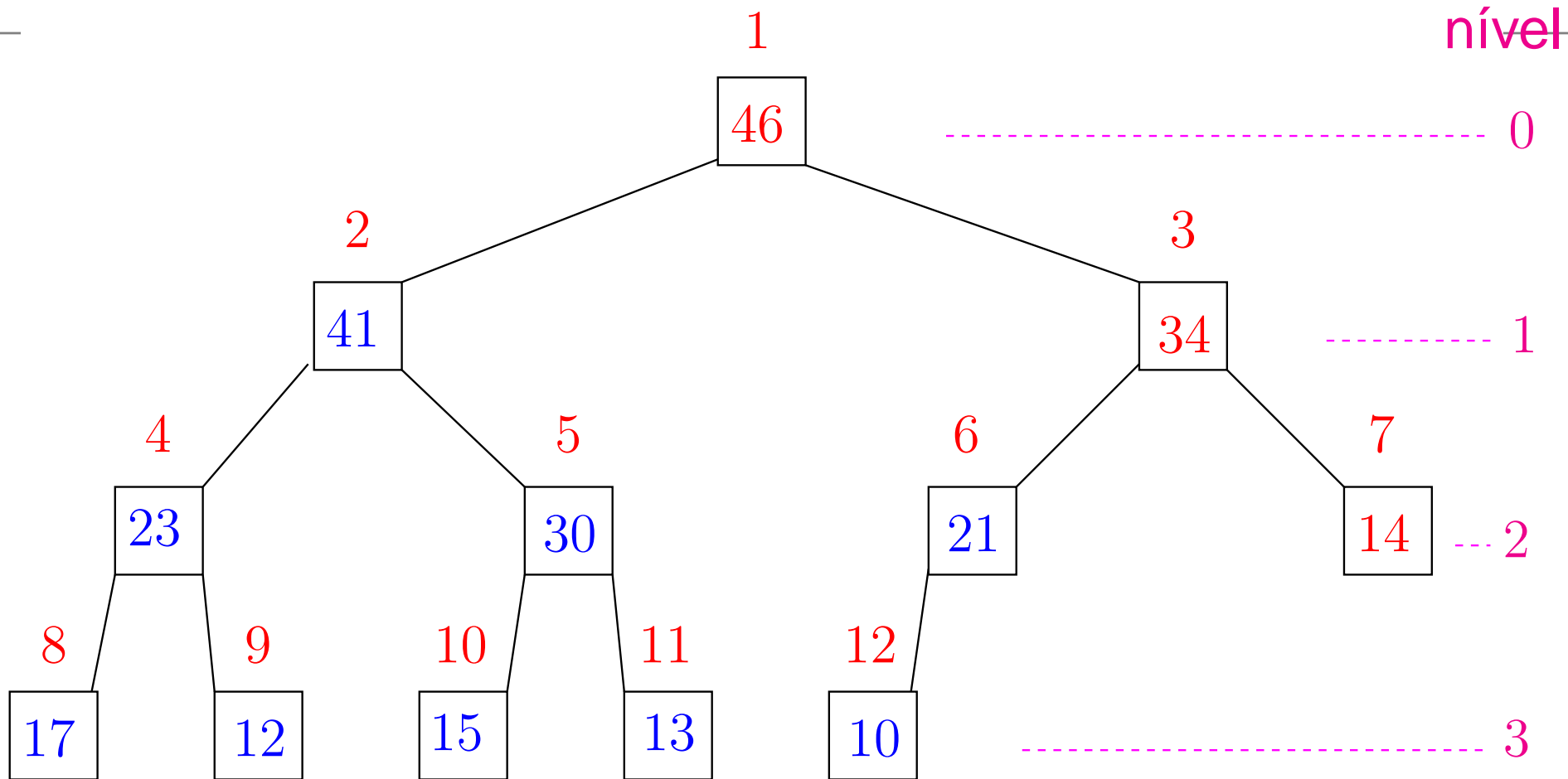
1	2	3	4	5	6	7	8	9	10	11	12
14	41	46	23	30	21	34	17	12	15	13	10

Construção de um heap



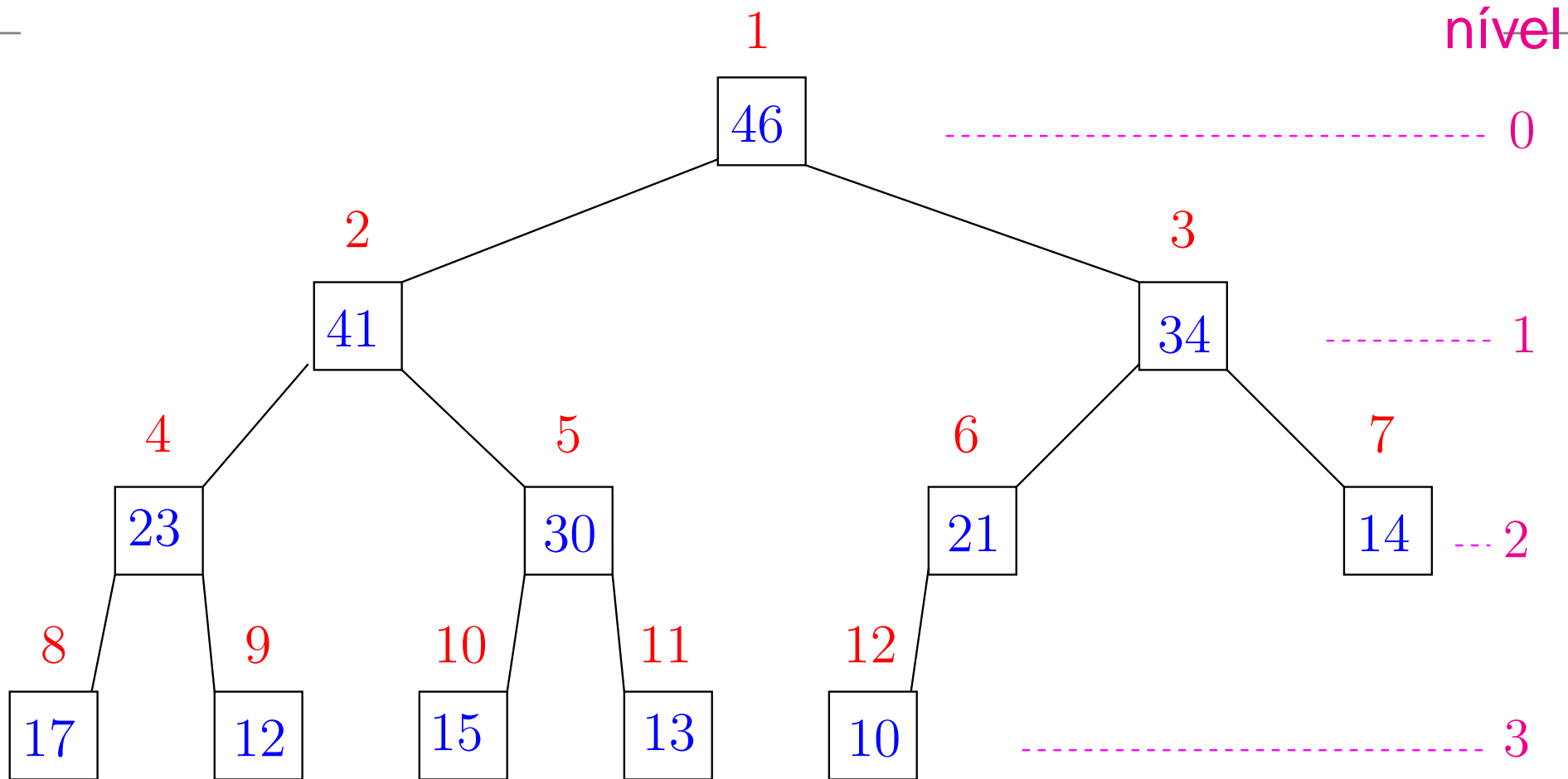
1	2	3	4	5	6	7	8	9	10	11	12
46	41	14	23	30	21	34	17	12	15	13	10

Construção de um heap



1	2	3	4	5	6	7	8	9	10	11	12
46	41	34	23	30	21	14	17	12	15	13	10

Construção de um heap



1	2	3	4	5	6	7	8	9	10	11	12
46	41	34	23	30	21	14	17	12	15	13	10

Construção de um heap

Recebe um vetor $A[1 \dots n]$ e **rearranja** A para que seja heap.

CONSTRÓI-HEAP (A, n)

2 **para** $i \leftarrow \lfloor n/2 \rfloor$ **decrecendo até** 1 **faça**

3 **DESCE-HEAP** (A, n, i)

Relação invariante:

(i0) no início de cada iteração, $i + 1, \dots, n$ são raízes de heaps.

$T(n) :=$ consumo de tempo no pior caso

Construção de um heap

Recebe um vetor $A[1..n]$ e rearranja A para que seja heap.

CONSTRÓI-HEAP (A, n)

2 para $i \leftarrow \lfloor n/2 \rfloor$ decrescendo até 1 faça

3 DESCE-HEAP (A, n, i)

Relação invariante:

(i0) no início de cada iteração, $i + 1, \dots, n$ são raízes de heaps.

$T(n)$:= consumo de tempo no pior caso

Análise grosseira: $T(n)$ é $\frac{n}{2} O(\lg n) = O(n \lg n)$.

Análise mais cuidadosa: $T(n)$ é ????.