

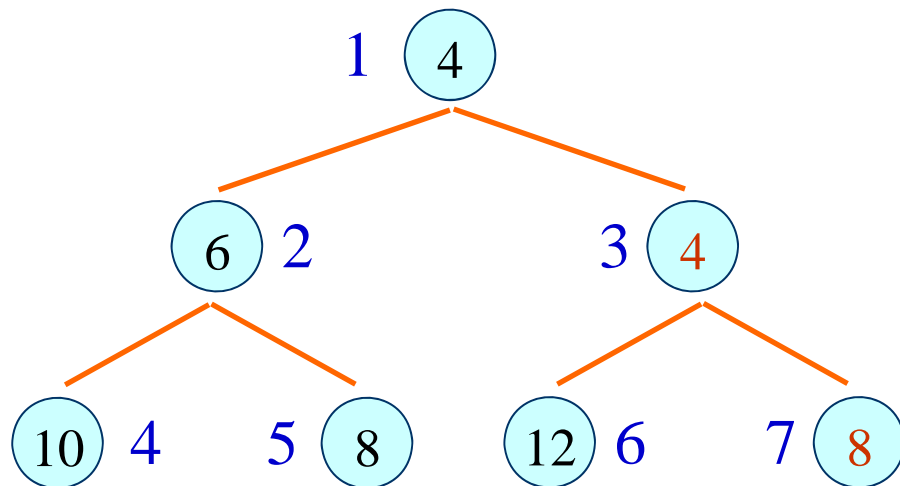
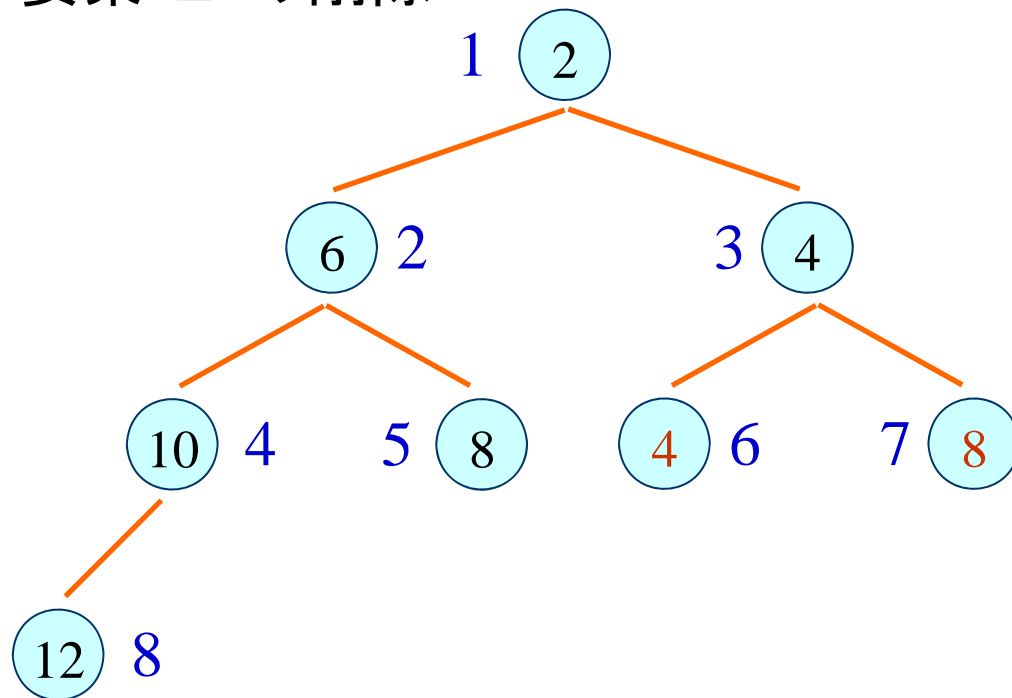
## 完全半順序二分木からの最小要素の削除

ヒープの先頭が、木の根であり最小要素。

- (1) ヒープの末尾の要素を、ヒープの先頭に移し、移動要素  $r$  とする。
- (2) 節点数  $n$  を 1 減らし、ヒープ末尾を 1 つ繰上げる。
- (3) 繰り返す（より下に対し局所的な半順序の保証）。
  - (3-1) 移動要素  $r$  が葉にあれば (\*), (3) の終り。
  - (3-2) 移動要素  $r$  が存在する節点を持つ、2 つの子の要素のうち、キー値が小さい要素を  $s$  とする。  
子が 1 つの場合は、その要素を  $s$  とする。
  - (3-3) 移動要素  $r$  と  $s$  でキーを比較し、移動要素のキーが大きければ、  
移動要素  $r$  と  $s$  を入れ替える。  
さもなければ、  
(3) の終り。

(\*) 葉であることは、 $(n/2$  (切捨て)  $+ 1) \sim n$  で判断。

# 最小要素 2 の削除



先頭 →

1		4
2		6
3		4
4		1 0
5		8
6		1 2
7		8
8		1 2
9		
10		

末尾 →

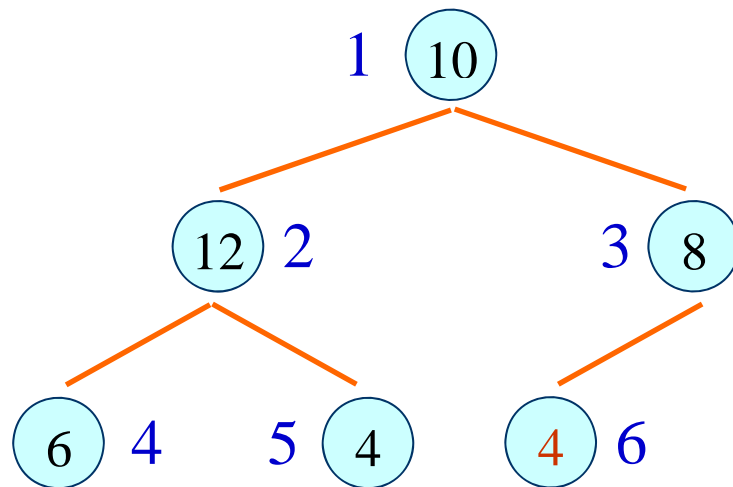
完全半順序 2 分木の性質は、失われない。

# ヒープによる完全二分木の、半順序木への変換

節点数を  $n$  とする.

(1)  $i$  を,  $n/2$  (切捨て) から 1 まで, 繰り返す.

(1-1)  $\text{node}[i]$  を移動要素  $r$  とし, 最小要素の削除の (3) を行う.

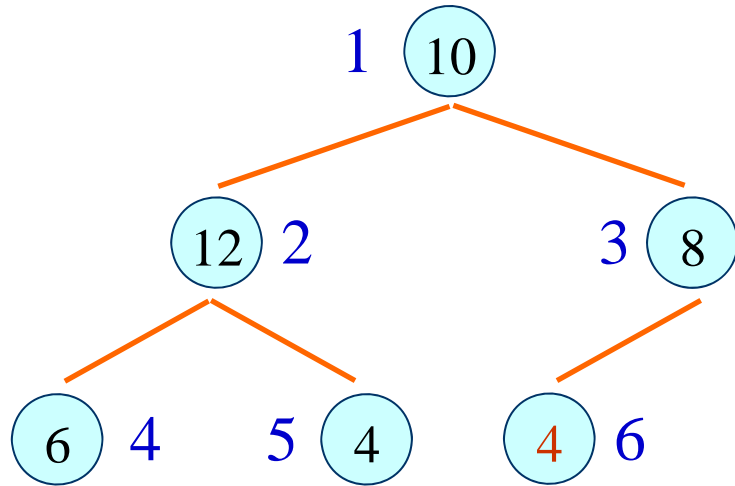


先頭 →

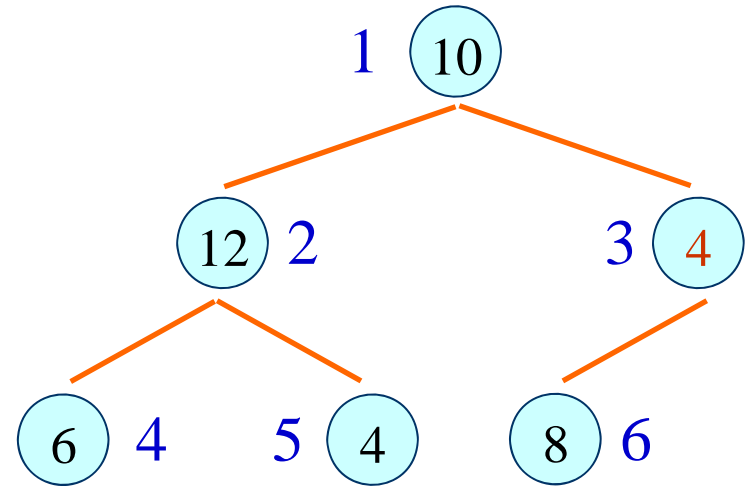
1		1 0
2		1 2
3		8
4		6
5		4
6		4

末尾 →

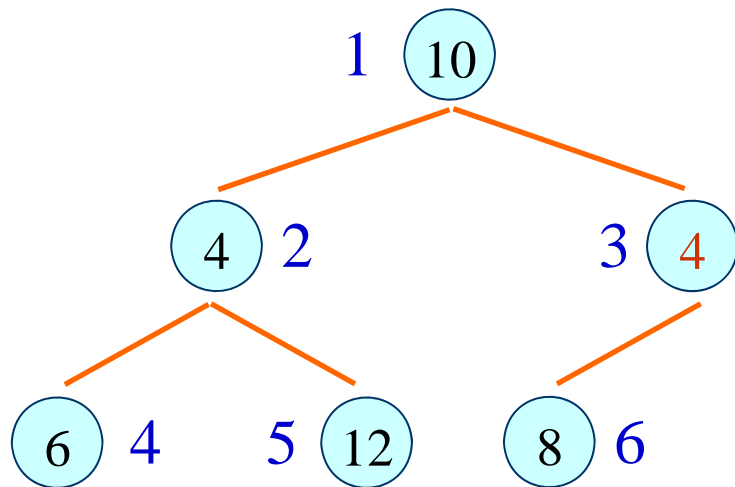
初期状態



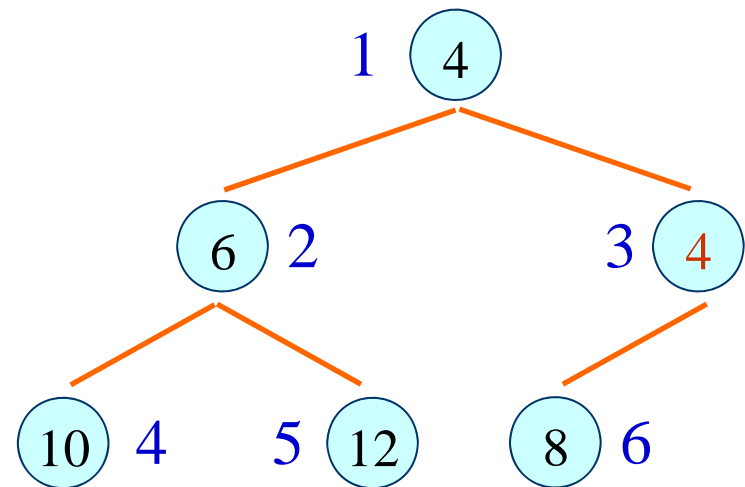
1回目終了時  $i = 3$



2回目終了時  $i = 2$



3回目終了時  $i = 1$



## 完全半順序 2 分木からの特定要素 $\text{node}[i]$ の削除

節点数を  $n$  とする.

- (1) ヒープの末尾の要素  $\text{node}[n]$  を, 節点  $i$  の位置に置く.
- (2) 節点数  $n$  を 1 減らし, ヒープ末尾を 1 つ繰上げる.
- (3) 現在, 節点  $i$  にある要素 (末尾にあった要素) を追加要素  $p$  として, 要素の追加の (3) を行う.
- (4) 現在, 節点  $i$  にある要素を移動要素  $r$  として, 最小要素の削除の (3) を行う.

## 処理の手間

追加, 削除とも, 処理の手間は, 木の深さに比例する.

完全半順序 2 分木の要素追加, (最小も含めた) 要素  
削除の計算量は,

$$O(\log_2(n))$$

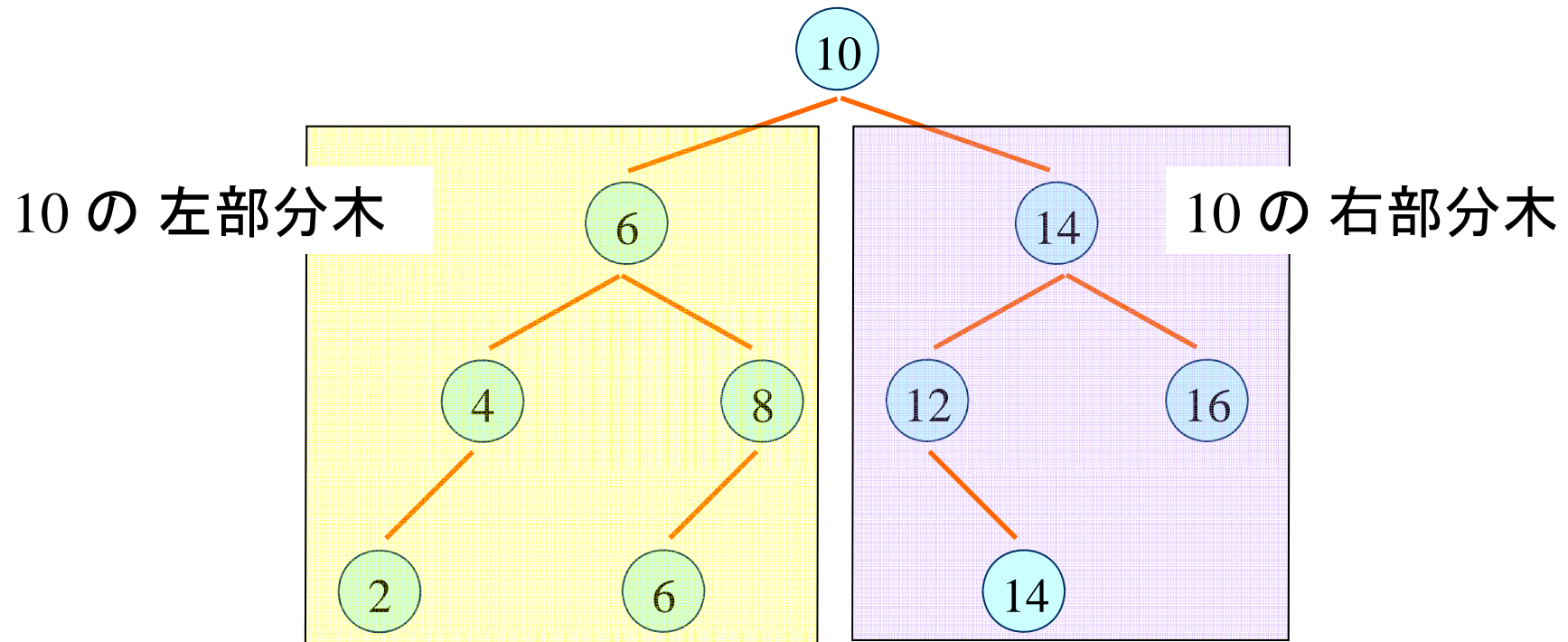
完全 2 分木の半順序木への変換は,

$$O(n \log_2(n))$$

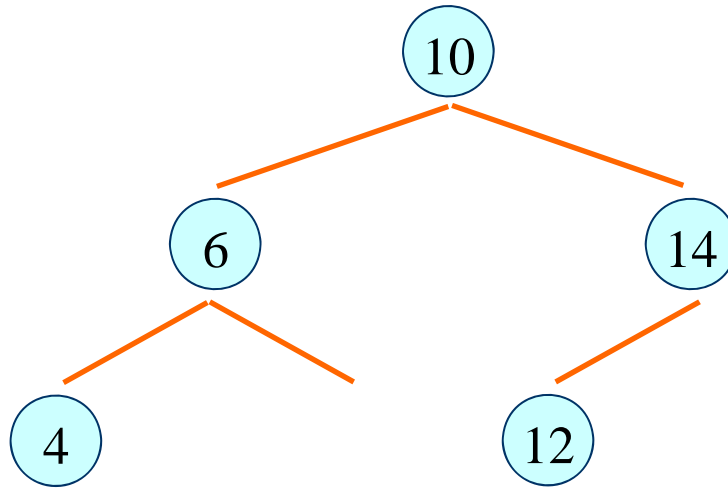
## その他の木

2分探索木 (完全2分木である必要はない)

2分木の節点  $i$  にある要素のキー値  $k_i$  に対し,  
左部分木にある要素のキー値は  $k_i$  より小さい.  
右部分木にある要素のキー値は  $k_i$  より大きい.



効率的な2分探索木  
高さ  $O(\log_2 n)$



非効率的な2分探索木  
高さ  $O(n)$

