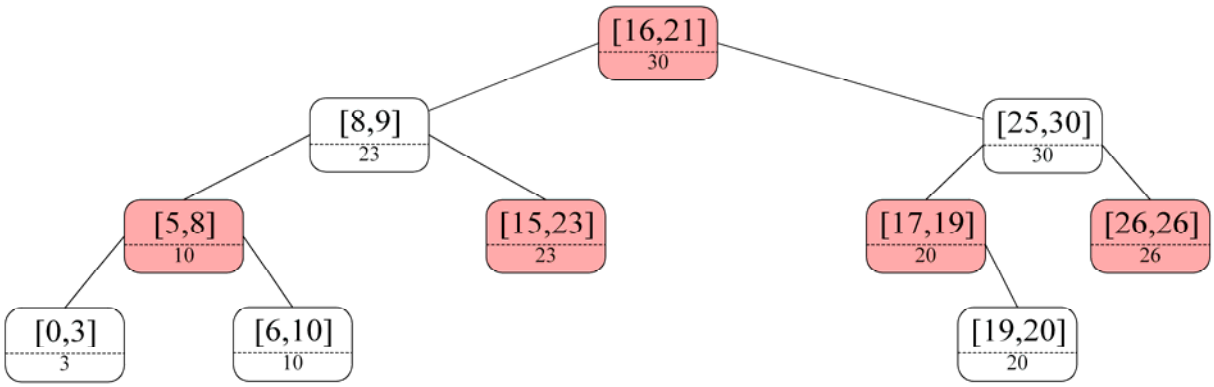
Übung 1.4.3: Intervallbäume

INTERVAL-SEARCH(T, i)
1 $x \leftarrow root[T]$
2 **while** $x \neq nil[T]$ and i und $int[x]$ überschneiden sich nicht
3 **do if** $left[x] \neq nil[T]$ and $max[left[x]] \geq low[i]$
4 **then** $x \leftarrow left[x]$
5 **else** $x \leftarrow right[x]$
6 **return** x

Aufgabe 1

Der Pseudocode INTERVAL-SEARCH(T, i) wählt in Abhängigkeit von dem Ergebnis der Abfrage in Zeile 3 für die weitere Suche nach einer Überlappung den linken oder den rechten Teilbaum des Knotens x.

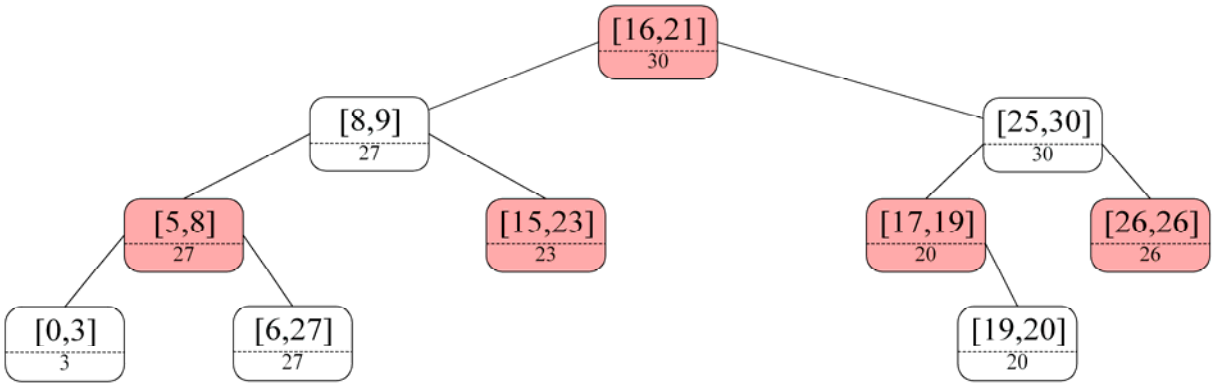


Unter welchen Bedingungen kann eine Intervallüberschneidung im linken und im rechten Teilbaum existieren? Diskutieren Sie diese Frage sowohl für die **erfolgreiche Suche**, z.B. i = [22, 27], als auch für die **erfolglose Suche**, z.B. i = [11, 14] von INTERVAL-SEARCH.

Aufgabe 2

Führen Sie die Operation INTERVAL-SEARCH(T, i) für den folgenden Baum T und das Intervall i = [25, 31] aus. Verwenden Sie beim Tracen den Pseudocode:

```
INTERVAL-SEARCH(T, i)
1    x ← root[T]
2    while x ≠ nil[T] and i und int[x] überschneiden sich nicht
3          do if left[x] ≠ nil[T] and max[left[x]] ≥ low[i]
4                then  x ← left[x]
5                else  x ← right[x]
6    return x
```



Notieren jeden Schritt des Algorithmus in der folgenden Tabelle.

Nr	**x**	**[25, 31] ∩ int[x]**	**left[x]**	**right[x]**	**max[left[x]] ≥ low[i] ?**
1	16	[25, 31] ∩ [16, 21] = ∅	8	25	27 ≥ 25 √
2					
3					
4					
5					
6					