

**Hilfsfunktion: Finde den ersten Knoten, der innerhalb des gegebenen Intervalls liegt — dort gabelt sich der Weg für eindimensionales orthogonal search.**

```
def find_split_node(binary_tree, x_min, x_max):
    # Beginne bei der Wurzel von binary_tree:
    v = binary_tree.root
    # Solange v kein Blatt ist und das Intervall (x_min, x_max)
    # "entweder links oder rechts" vom aktuellen Knoten v liegt ...
    while not v.is_leaf() and (x_max <= v.key or x_min > v.key):
        # ... setze mit "richtigem Teilbaum" fort:
        if x_max <= v.key:
            v = v.left
        else:
            v = v.right
    # Ok: v ist entweder ein Blatt, oder x_min <= v.key < x_max
    return v
```