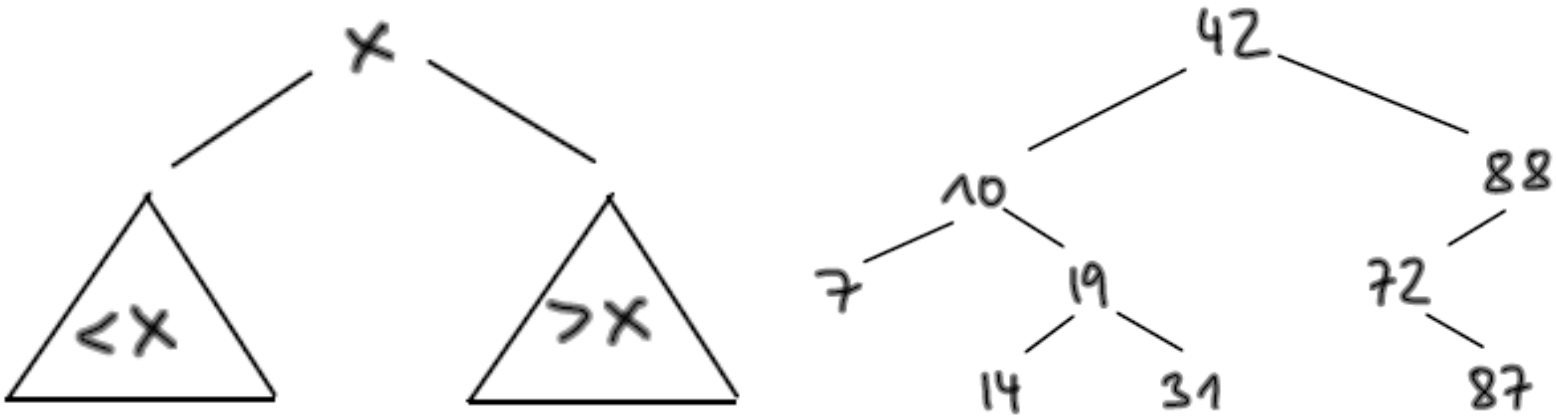
Informatik

Suchbaum

Ein *binärer Suchbaum* ist ein binärer Baum, bei dem alle Einträge im linken Teilbaum eines Knotens x kleiner sind als der Eintrag im Knoten x und bei dem alle Einträge im rechten Teilbaum eines Knotens x größer sind als der Eintrag im Knoten x.

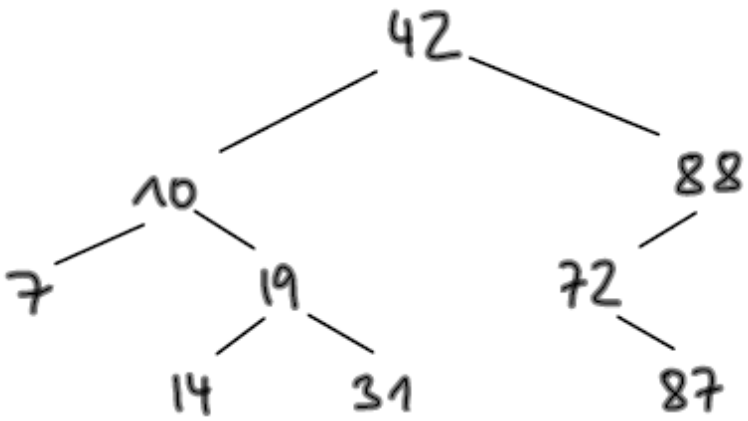


Methoden des Suchbaums:

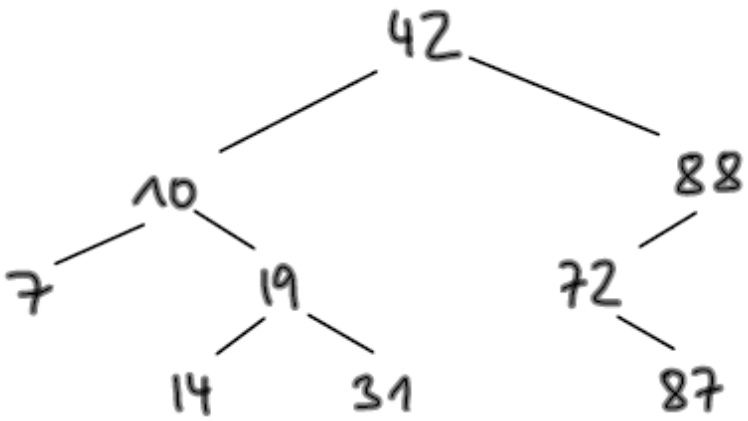
lookup(x): zielgerichtet absteigen und nach x schauen
falls gefunden: Verweis liefern, falls nicht: melden, dass nicht vorhanden

insert(x): zielgerichtet absteigen und nach x schauen
falls gefunden: melden, dass insert nicht möglich,
falls nicht gefunden: dort einhängen

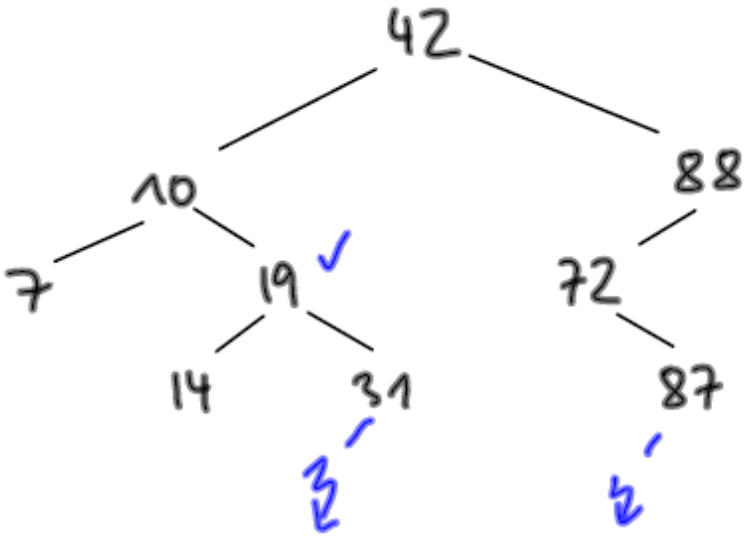
delete(x): zielgerichtet absteigen und nach x schauen
falls gefunden: entfernen,
falls nicht gefunden: melden, dass delete nicht möglich

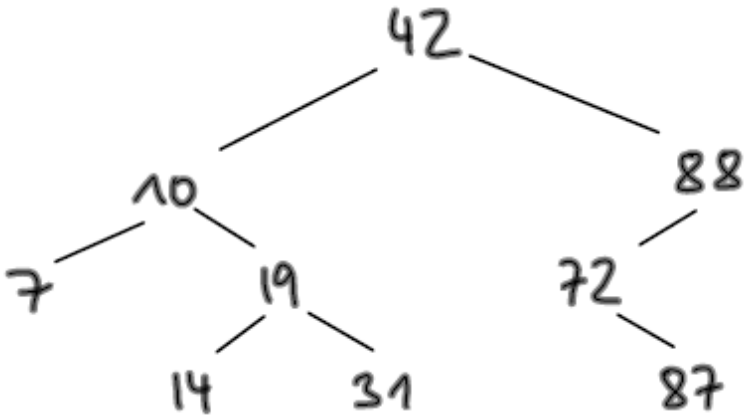


lookup 20, 19, 85

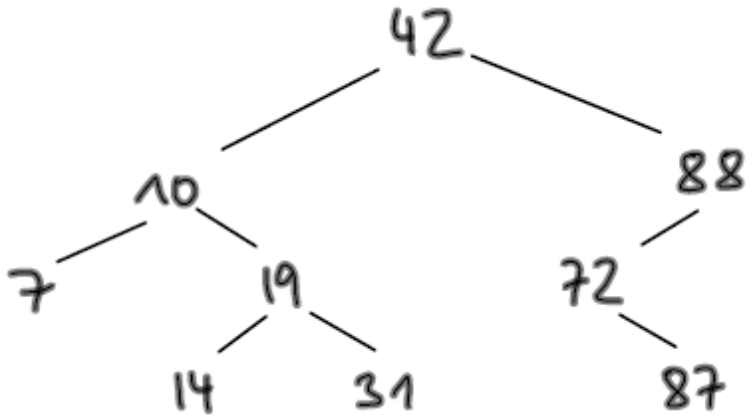


lookup 20, 19, 85

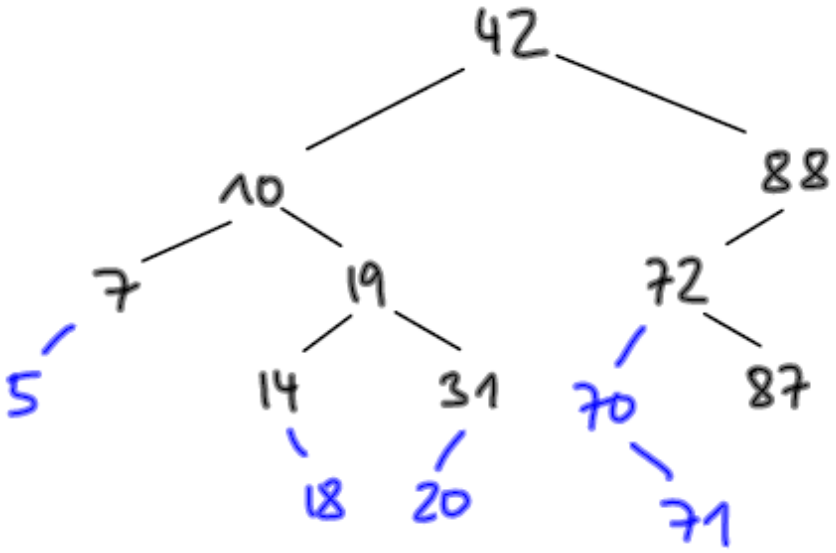




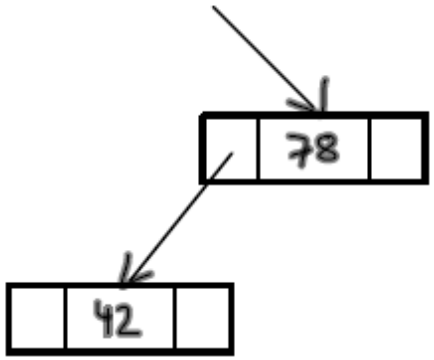
insert 5, 18, 70, 71, 20



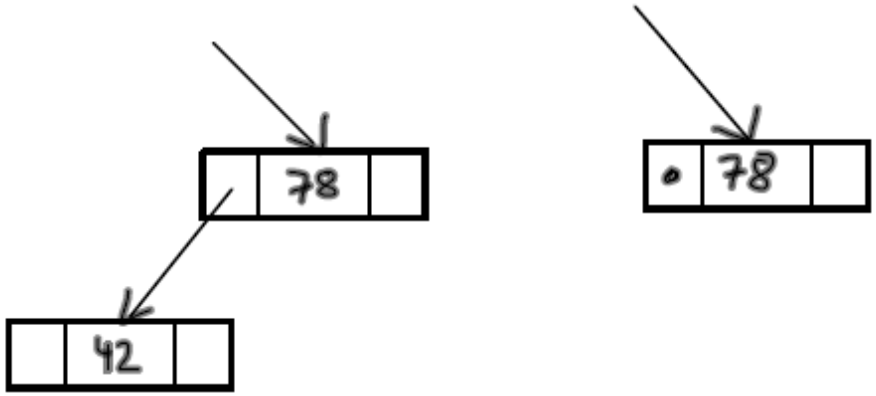
insert 5, 18, 70, 71, 20



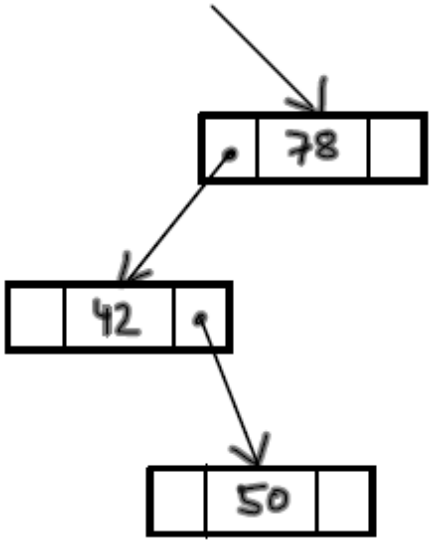
delete 42 - Fall 1: Blatt löschen



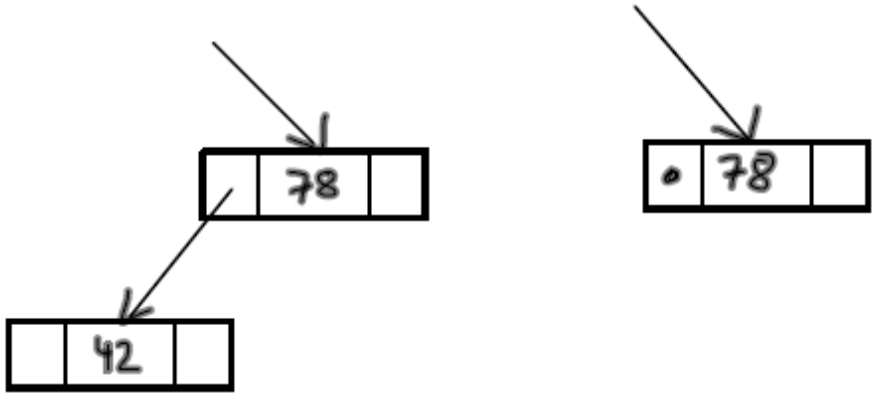
delete 42 - Fall 1: Blatt löschen



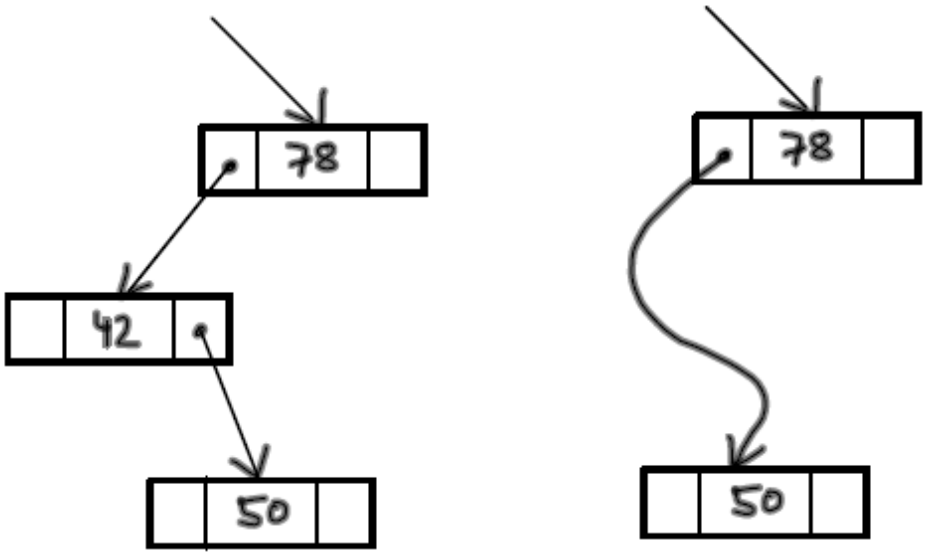
delete 42 - Fall 2: ein Sohn



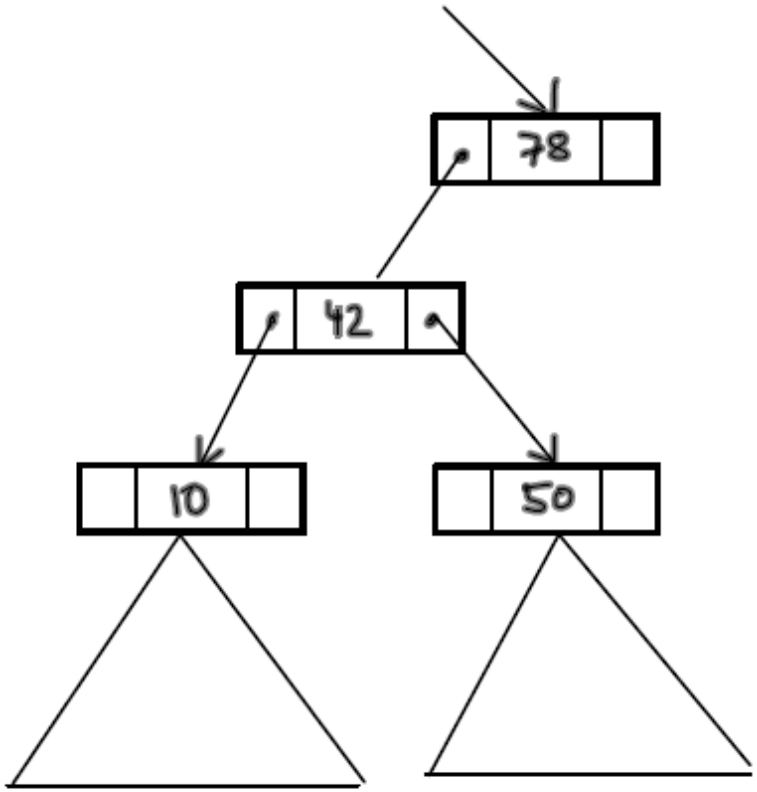
delete 42 - Fall 1: Blatt löschen



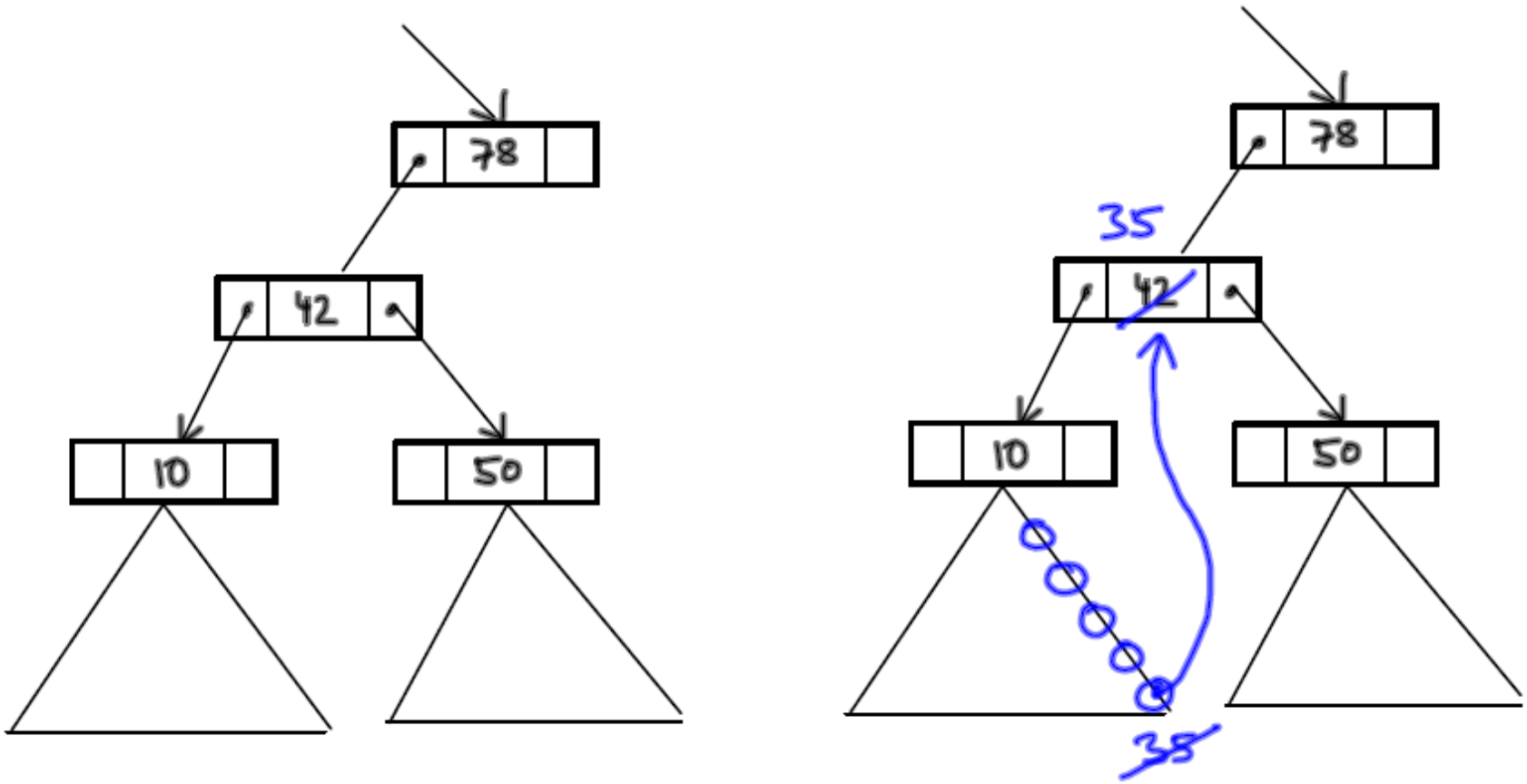
delete 42 - Fall 2: ein Sohn



delete 42 - Fall 3: zwei Söhne



delete 42 - Fall 3: zwei Söhne



Löschen der 35: Fall 1 oder Fall 2.

Komplexität der Methoden im Suchbaum: Anzahl Ebenen

worst case:

Komplexität der Methoden im Suchbaum: Anzahl Ebenen

worst case: 

O(n)

best case: 

Komplexität der Methoden im Suchbaum: Anzahl Ebenen

worst case: 

O(n)

best case:  O(log(n))

average case: O(log(n))

```
"""
lookup:
Beginne bei der Wurzel
Solange ein Knoten vorliegt
      vergleiche x mit Knoteninhalt
      steige je nach Vergleich in den linken oder
          rechten Teilbaum ab oder liefere bei
          Gleichheit das Objekt zurück
Liefere null zurück
"""
```

```
"""
lookup:
Beginne bei der Wurzel
Solange ein Knoten vorliegt
      vergleiche x mit Knoteninhalt
      steige je nach Vergleich in den linken oder
         rechten Teilbaum ab oder liefere bei
         Gleichheit das Objekt zurück
Liefere null zurück
"""
def lookup(self,x):
    k = self.wurzel
    while k is not None:
        if x < k.inhalt:
            k = k.links
        elif x > k.inhalt:
            k = k.rechts
        else:
            return k.inhalt
    return None
```

```
"""
insert:
wenn Baum leer
      füge neuen Knoten mit x ein
      melde Erfolg
sonst
      beginne bei der Wurzel
      solange ein Knoten vorliegt
            merke aktuellen Knoten als Vater
            steige je nach Vergleich in den linken oder
                  rechten Teilbaum ab oder
                  melde bei Gleichheit Mißerfolg
      falls x kleiner als vater
            hänge neuen Knoten links an
      sonst
            hänge neuen Knoten rechts an
      melde Erfolg
"""
```

```
def insert(self,x):
    if self.wurzel is None:
        self.wurzel = Knoten(x)
        return True
    else:
        vater = None
        k = self.wurzel
        while k is not None:
            vater = k
            if (x < k.inhalt):
                k = k.links
            elif (x > k.inhalt):
                k = k.rechts
            else:
                return False
        if x < vater.inhalt:
            vater.links = Knoten(x)
        else:
            vater.rechts = Knoten(x)
        return True
```

Objekte in einen Suchbaum stopfen:

Ist in einer Klasse die magische Methode `__lt__(self,other)` implementiert, dann wird beim Vergleich mit dem Zeichen < diese Methode aufgerufen.

```
class Person:
    def __init__(self,name,alter):
        self.name = name
        self.alter = alter

    def __lt__(self,other):
        return self.name < other.name
```