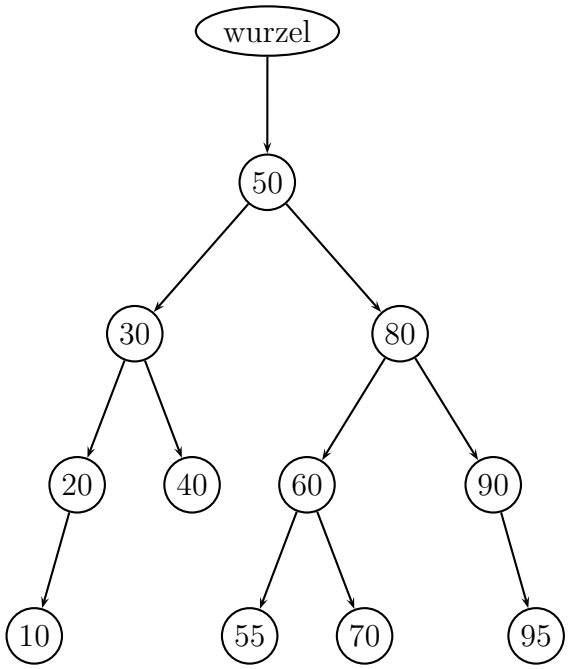
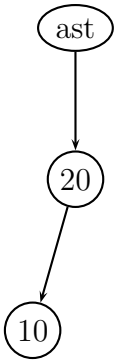
1 Löschen in einem geordneten Binärbaum

Der dargestellte Baum sei bereits aufgebaut. Nun sollen Elemente in diesem Baum gelöscht werden. Dabei ergeben sich verschiedene Fälle.

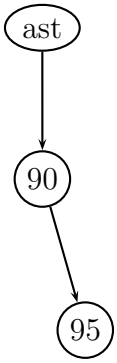


Fall 1: Ein *Blatt* oder *Endknoten* ist zu löschen, z.B. hier der Knoten mit Inhalt 10.

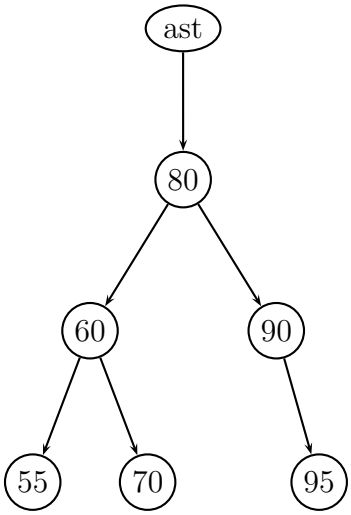
Fall 2: Ein *innerer Knoten* mit einem linken, aber keinem rechten Sohn ist zu löschen, z.b. hier der Knoten mit dem Inhalt 20. Achtung: Der linke Sohn kann ein Teilbaum sein!



Fall 3: Ein *innerer Knoten* mit einem rechten, aber keinem linken Sohn ist zu löschen, z.b. hier der Knoten mit dem Inhalt 90. Achtung: Der rechte Sohn kann ein Teilbaum sein!



Fall 4: Ein *innerer Knoten* mit einem rechten **und** einem linken Sohn ist zu löschen, z.b. hier der Knoten mit dem Inhalt 80. Achtung: Der linke und der rechte Sohn können große Teilbäume sein!



Aufgaben:

1. Zeichne für jeden Fall die nötigen Operationen zur Löschung mit Hilfe von geeigneten Zeigermodifikationen.
2. Welche Fälle lassen sich weitgehend zusammenfassend behandeln?
3. Formuliere in Worten, wie man den letzten und kompliziertesten Fall geeignet behandeln kann.

2 Durchlaufen eines geordneten Binärbaums

Der oben dargestellte Baum wird auf verschiedene Arten durchlaufen. Stelle für jeden Fall die Ausgabe dar:

1. Mit *inOrder*:
2. Mit *preOrder*:
3. Mit *postOrder*:

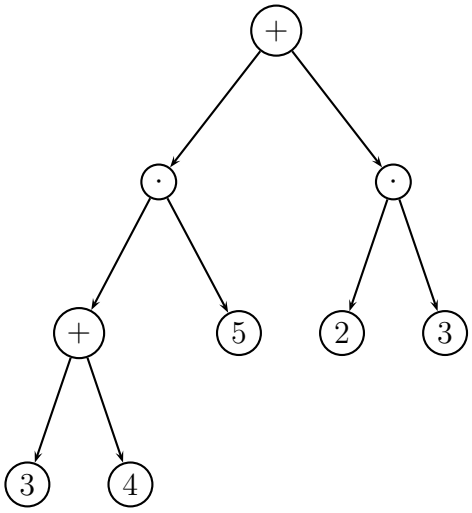
```
class BinBaum
{ BinBaum links;   // linker Teilbaum
  BinBaum rechts;  // rechter Teilbaum
  int inhalt;      // Inhalt soll int sein.

  // Konstruktor
  BinBaum (int zahl)
  { this.links  = null;    this.rechts = null;    this.inhalt = zahl;
  }
  ...
  void inOrder ()
  {  if (links != null) links.inOrder();
     Out.print(inhalt + "  ");
     if (rechts != null) rechts.inOrder();
  } // inOrder

  void preOrder ()
  {  Out.print(inhalt + "  ");
     if (links != null) links.preOrder();
     if (rechts != null) rechts.preOrder();
  } // preorder

  void postOrder ()
  {  if (links != null) links.postOrder();
     if (rechts != null) rechts.postOrder();
     Out.print(inhalt + "  ");
  } // postOrder
} // class BinBaum
```

Hinweis: An Stelle der Methode `print` könnte natürlich irgendeine Methode stehen, die etwas mit den Inhalten macht.

Beispiel 1:
Der unten dargestellte Baum ist eine Repräsentation des arithmetischen Ausdrucks

$$(3 + 4) \cdot 5 + 2 \cdot 3.$$

1. Wie muss der Baum durchlaufen werden, um aus dem Baum den arithmetischen Ausdruck (mit geeigneten Klammerungen) auszulesen?
2. Wie muss der Baum durchlaufen werden, um aus dem Baum den arithmetischen Ausdruck durch Überführung in einen Stack mit UPN zu berechnen?



Beispiel 2:
In der baumartigen Verzeichnisstruktur eines Dateisystems soll der Platzbedarf auf einem Datenträger ermittelt werden. Nach welchem Verfahren muss die Dateistruktur durchlaufen werden?

Beispiel 3:
Eine baumartige Verzeichnisstruktur eines Dateisystems soll durch Einrückung dargestellt werden. Nach welchem Verfahren muss die Dateistruktur durchlaufen werden?