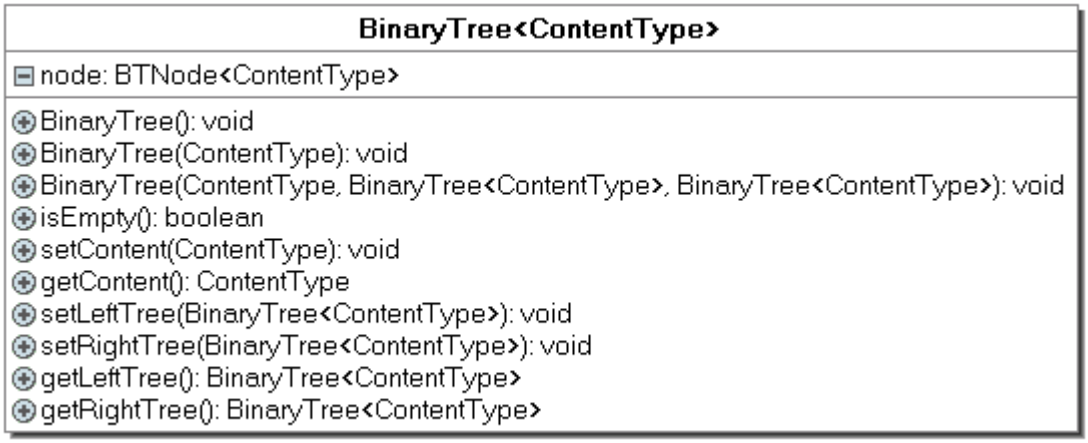
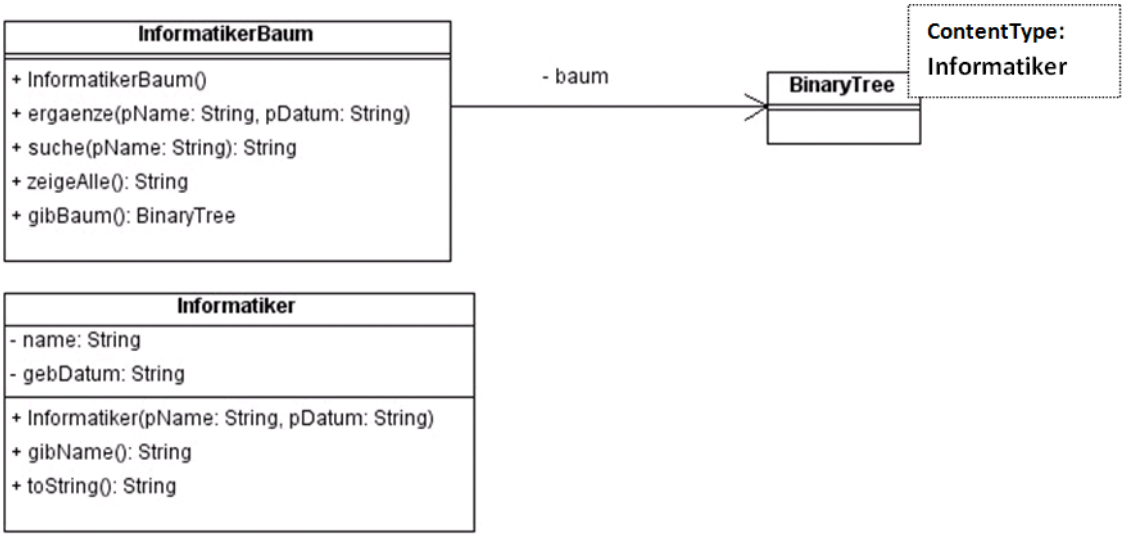
Informatiker-Baum

Der allgemeine Binärbaum
Es wird eine Baumstruktur entworfen und benutzt, bei der beliebige Inhaltsobjekte abgespeichert werden können. Eine Klasse, die diese Aufgabe übernehmen kann, ist die Klasse **`BinaryTree`** (ZA-Klasse). Im Folgenden wird das sortierte Einfügen in den Binärbaum, das Suchen im Binärbaum und die Ausgabe aller Daten der Inhaltsobjekte in sortierter Reihenfolge realisiert.

Die Klasse **`BinaryTree`**<**`ContentType`**>
Mithilfe der generischen Klasse **`BinaryTree` können** beliebig viele Objekte vom Typ **`ContentType`** in einem Binärbaum verwaltet werden. Ein Objekt der Klasse stellt entweder einen leeren Baum dar oder verwaltet ein Inhaltsobjekt sowie einen linken und einen rechten Teilbaum, die ebenfalls Objekte der generischen Klasse **`BinaryTree`** sind.



Realisierung des Informatiker-Baums mithilfe der Klasse **`BinaryTree`**



Der Informatiker-Baum ist nach den Namen alphabetisch sortiert. Jeder Informatiker wird nur einmal in den Baum eingefügt.

Einfügen in den Informatiker-Baum

```
19
20  private void fuegeEin(BinaryTree<Informatiker> b, String pName, String pDatum) {
21    if (b.isEmpty()) {
22      Informatiker neu = new Informatiker(pName, pDatum);
23      b.setContent(neu);
24    } else {
25      Informatiker inf = b.getContent();
26      if (pName.compareTo(inf.gibName()) < 0) {
27        fuegeEin(b.getLeftTree(),pName,pDatum);
28      } else {
29        if (pName.compareTo(inf.gibName()) > 0) {
30          fuegeEin(b.getRightTree(),pName,pDatum);
31        } else {
32          System.out.println("Name bereits vorhanden");
33        }
34      }
35    }
36  }
```

Suchen im Informatiker-Baum

Der Baum ist nach Namen sortiert. Durch Vergleich des gegebenen Namens mit dem Namen im Inhaltsobjekt des Teilbaums, kann der passende Suchweg im Baum gefunden werden.

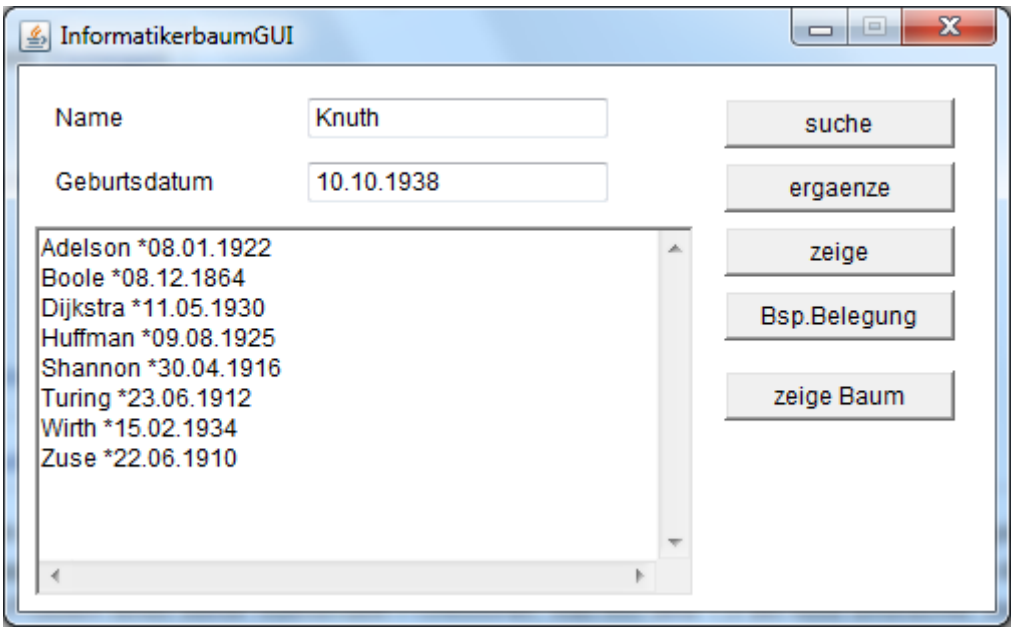
```
37
38  public String suche(String pName) {
39    return baumSuche(baum,pName);
40  }
41
42  private String baumSuche(BinaryTree<Informatiker> b, String pName) {
43    if (b.isEmpty()) {
44      return "--";
45    } else {
46      Informatiker inf = b.getContent();
47      if (pName.compareTo(inf.gibName()) < 0) {
48        return baumSuche(b.getLeftTree(),pName);
49      } else {
50        if (pName.compareTo(inf.gibName()) > 0) {
51          return baumSuche(b.getRightTree(),pName);
52        } else {
53          return inf.toString();
54        }
55      }
56    }
57  }
```

Sortierte Ausgabe

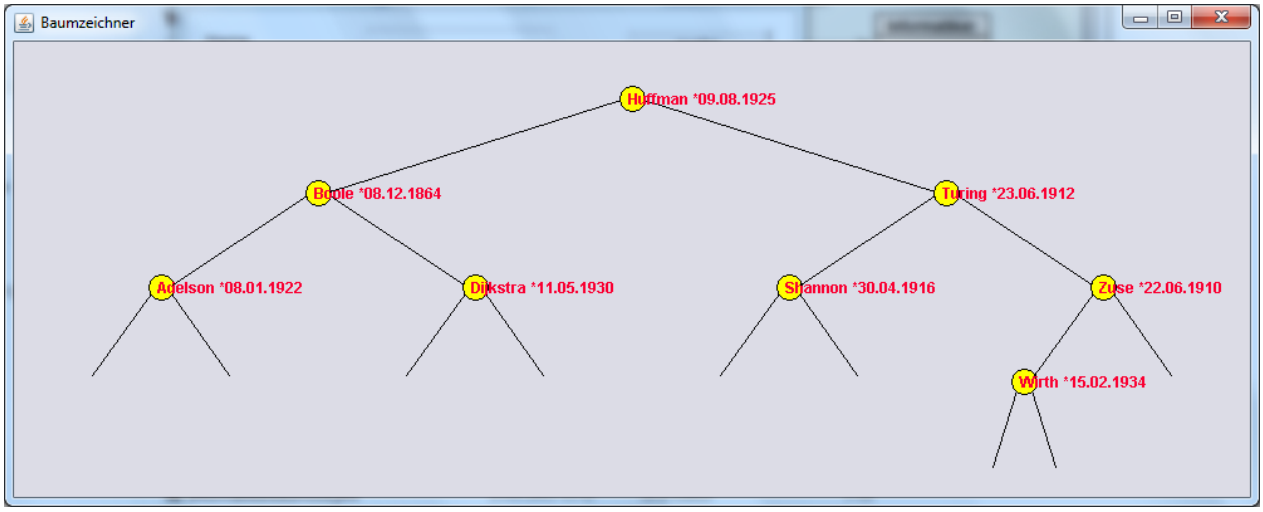
Der Baum wird nach der inorder-Strategie durchlaufen und von jedem Inhaltsobjekt des Teilbaums wird der Name und das Geburtsdatum ausgegeben.

```
58
59    public String zeigeAlle() {
60      return durchlaufe(baum);
61    }
62
63    private String durchlaufe(BinaryTree<Informatiker> b) {
64      String aktuell = "";
65      String links = "";
66      String rechts = "";
67      if (!b.isEmpty()) {
68        links = durchlaufe(b.getLeftTree());
69        aktuell = (b.getContent()).toString() + "\n" ;
70        rechts = durchlaufe(b.getRightTree());
71      }
72      return links + aktuell + rechts ;
73    }
74
```

Benutzungsoberfläche (mit zusätzlichen, schon implementierten Möglichkeiten, die das Testen erleichtern)
Die im Informatiker-Baum gespeicherten Daten (Name, Geburtsdatum) werden in sortierter Reihenfolge ausgegeben.



Zusatz: Darstellung des Binärbaums
In einem seperaten Fenster wird der Binärbaum ausgegeben.


Baumzeichner
Huffman *09.08.1925
Boole *08.12.1864
Turing *23.06.1912
Adelson *08.01.1922
Dijkstra *11.05.1930
Shannon *30.04.1916
Zuse *22.06.1910
Wirth *15.02.1934