Die Klasse `BinarySearchTree<ContentType extends ComparableContent<ContentType>>`

Mithilfe der generischen Klasse **`BinarySearchTree`** können beliebig viele Objekte des Typs **`ContentType`** in einem Binärbaum (binärer Suchbaum) entsprechend einer Ordnungsrelation verwaltet werden.
Ein Objekt der Klasse **`BinarySearchTree`** stellt entweder einen leeren Baum dar oder verwaltet ein Inhaltsobjekt vom Typ **`ContentType`** sowie einen linken und einen rechten Teilbaum, die ebenfalls Objekte der Klasse **`BinarySearchTree`** sind.
Die Klasse der Objekte, die in dem Suchbaum verwaltet werden sollen, muss das generische Interface **`ComparableContent`** implementieren. Dabei muss durch Überschreiben der drei Vergleichsmethoden `isLess, isEqual, isGreater` (s. Dokumentation des Interfaces) eine eindeutige Ordnungsrelation festgelegt sein.

Beispiel einer solchen Klasse:

```
public class Entry implements ComparableContent<Entry> {

  int wert;
  // diverse weitere Attribute

  public boolean isLess(Entry pContent) {
    return this.getWert() < pContent.getWert();
  }

  public boolean isEqual(Entry pContent) {
    return this.getWert() == pContent.getWert();
  }

  public boolean isGreater(Entry pContent) {
    return this.getWert() > pContent.getWert();
  }

  public int getWert() {
    return this.wert;
  }
```

Die Objekte der Klasse **`ContentType`** sind damit vollständig geordnet. Für je zwei Objekte c1 und c2 vom Typ **ContentType** gilt also insbesondere genau eine der drei Aussagen:

- `c1.isLess(c2)` (Sprechweise: c1 ist kleiner als c2)
- `c1.isEqual(c2)` (Sprechweise: c1 ist gleichgroß wie c2)
- `c1.isGreater(c2)` (Sprechweise: c1 ist größer als c2)

Alle Objekte im linken Teilbaum sind kleiner als das Inhaltsobjekt des Binärbaumes. Alle Objekte im rechten Teilbaum sind größer als das Inhaltsobjekt des Binärbaumes.
Diese Bedingung gilt auch in beiden Teilbäumen.

Dokumentation der generischen Klasse `BinarySearchTree<ContentType extends ComparableContent<ContentType>>`

Konstruktor **`BinarySearchTree()`**
Der Konstruktor erzeugt einen leeren Suchbaum.

Anfrage **`boolean isEmpty()`**
Diese Anfrage liefert den Wahrheitswert `true`, wenn der Suchbaum leer ist, sonst liefert sie den Wert `false`.

Auftrag **`void insert(ContentType pContent)`**
Falls bereits ein Objekt in dem Suchbaum vorhanden ist, das gleichgroß ist wie `pContent`, passiert nichts.
Andernfalls wird das Objekt `pContent` entsprechend der Ordnungsrelation in den Baum eingeordnet. Falls der Parameter `null` ist, ändert sich nichts.

Anfrage **`ContentType search(ContentType pContent)`**
Falls ein Objekt im binären Suchbaum enthalten ist, das gleichgroß ist wie `pContent`, liefert die Anfrage dieses, ansonsten wird `null` zurückgegeben. Falls der Parameter `null` ist, wird `null` zurückgegeben.

Auftrag **`void remove(ContentType pContent)`**
Falls ein Objekt im binären Suchbaum enthalten ist, das gleichgroß ist wie `pContent`, wird dieses entfernt. Falls der Parameter `null` ist, ändert sich nichts.

Anfrage **`ContentType getContent()`**
Diese Anfrage liefert das Inhaltsobjekt des Suchbaumes. Wenn der Suchbaum leer ist, wird `null` zurückgegeben.

Anfrage **`BinarySearchTree<ContentType> getLeftTree()`**
Diese Anfrage liefert den linken Teilbaum des binären Suchbaumes. Der binäre Suchbaum ändert sich nicht. Wenn er leer ist, wird `null` zurückgegeben.

Anfrage **`BinarySearchTree<ContentType> getRightTree()`**
Diese Anfrage liefert den rechten Teilbaum des Suchbaumes. Der Suchbaum ändert sich nicht. Wenn er leer ist, wird `null` zurückgegeben.

Das generische Interface (Schnittstelle) `ComparableContent<ContentType>`

Das generische Interface **`ComparableContent`** muss von Klassen implementiert werden, deren Objekte in einen Suchbaum (**`BinarySearchTree`**) eingefügt werden sollen. Die Ordnungsrelation wird in diesen Klassen durch Überschreiben der drei implizit abstrakten Methoden `isGreater`, `isEqual` und `isLess` festgelegt.

Das Interface **`ComparableContent`** gibt folgende implizit abstrakte Methoden vor:

Anfrage **`boolean isGreater(ContentType pComparableContent)`**

Wenn festgestellt wird, dass das Objekt, von dem die Methode aufgerufen wird, bzgl. der gewünschten Ordnungsrelation größer als das Objekt `pComparableContent` ist, wird `true` geliefert. Sonst wird `false` geliefert.

Anfrage **`boolean isEqual(ContentType pComparableContent)`**

Wenn festgestellt wird, dass das Objekt, von dem die Methode aufgerufen wird, bzgl. der gewünschten Ordnungsrelation gleich dem Objekt `pComparableContent` ist, wird `true` geliefert. Sonst wird `false` geliefert.

Anfrage **`boolean isLess(ContentType pComparableContent)`**

Wenn festgestellt wird, dass das Objekt, von dem die Methode aufgerufen wird, bzgl. der gewünschten Ordnungsrelation kleiner als das Objekt `pComparableContent` ist, wird `true` geliefert. Sonst wird `false` geliefert.