Die Klasse *`Item`*

Die Klasse *`Item`* ist abstrakte Oberklasse aller Klassen, deren Objekte in einen Suchbaum (`BinarySearchTree`) eingefügt werden sollen. Die Ordnungsrelation wird in den Unterklassen von `Item` durch Überschreiben der drei abstrakten Methoden `isEqual`, `isGreater` und `isLess` festgelegt.

Die Klasse `Item` gibt folgende abstrakte Methoden vor:

`abstract boolean isEqual(Item pItem)`
Wenn festgestellt wird, dass das Objekt, von dem die Methode aufgerufen wird, bzgl. der gewünschten Ordnungsrelation gleich dem Objekt `pItem` ist, wird `true` geliefert. Sonst wird `false` geliefert.

`abstract boolean isLess(Item pItem)`
Wenn festgestellt wird, dass das Objekt, von dem die Methode aufgerufen wird, bzgl. der gewünschten Ordnungsrelation kleiner als das Objekt `pItem` ist, wird `true` geliefert. Sonst wird `false` geliefert.

`abstract boolean isGreater(Item pItem)`
Wenn festgestellt wird, dass das Objekt, von dem die Methode aufgerufen wird, bzgl. der gewünschten Ordnungsrelation größer als das Objekt `pItem` ist, wird `true` geliefert. Sonst wird `false` geliefert.

Die Klasse BinarySearchTree

In einem Objekt der Klasse `BinarySearchTree` werden beliebig viele Objekte in einem Binärbaum (binärer Suchbaum) entsprechend einer Ordnungsrelation verwaltet. 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 Klasse `BinarySearchTree` sind. Dabei gilt:
Die Inhaltsobjekte sind Objekte einer Unterklasse von `Item`, in der durch Überschreiben der drei Vergleichsmethoden `isLess`, `isEqual`, `isGreater` (s. Item) eine eindeutige Ordnungsrelation festgelegt sein muss.
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.
Die Klasse `BinarySearchTree` ist keine Unterklasse der Klasse `BinaryTree`, sodass deren Methoden nicht zur Verfügung stehen.

Dokumentation der Klasse BinarySearchTree

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(Item pItem)`**
Falls ein bezüglich der verwendeten Vergleichsmethode `isEqual` mit `pItem` übereinstimmendes Objekt im geordneten Baum enthalten ist, passiert nichts. Andernfalls wird das Objekt `pItem` entsprechend der vorgegebenen Ordnungsrelation in den Baum eingeordnet. Falls der Parameter `null` ist, ändert sich nichts.

Anfrage **`Item search(Item pItem)`**
Falls ein bezüglich der verwendeten Vergleichsmethode `isEqual` mit `pItem` übereinstimmendes Objekt im binären Suchbaum enthalten ist, liefert die Anfrage dieses, ansonsten wird `null` zurückgegeben. Falls der Parameter `null` ist, wird `null` zurückgegeben.

Auftrag **`void remove(Item pItem)`**
Falls ein bezüglich der verwendeten Vergleichsmethode `isEqual` mit `pItem` übereinstimmendes Objekt im binären Suchbaum enthalten ist, wird dieses entfernt. Falls der Parameter `null` ist, ändert sich nichts.

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

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