

Übung 3: Binärer Suchbaum

Abgabetermin: 10.04.2018

Name:

Matrikelnummer:

Gruppe: G1 Di 10:15-11:00

~~G2 Di 11:00-11:45~~

G3 Di 10:15-11:00

Aufgabe	Punkte	gelöst	abzugeben schriftlich	abzugeben elektronisch	Korr.	Punkte
Aufgabe 1	24	<input type="checkbox"/>	Java-Programm Zeichensequenz	Projekt Archiv	<input type="checkbox"/>	

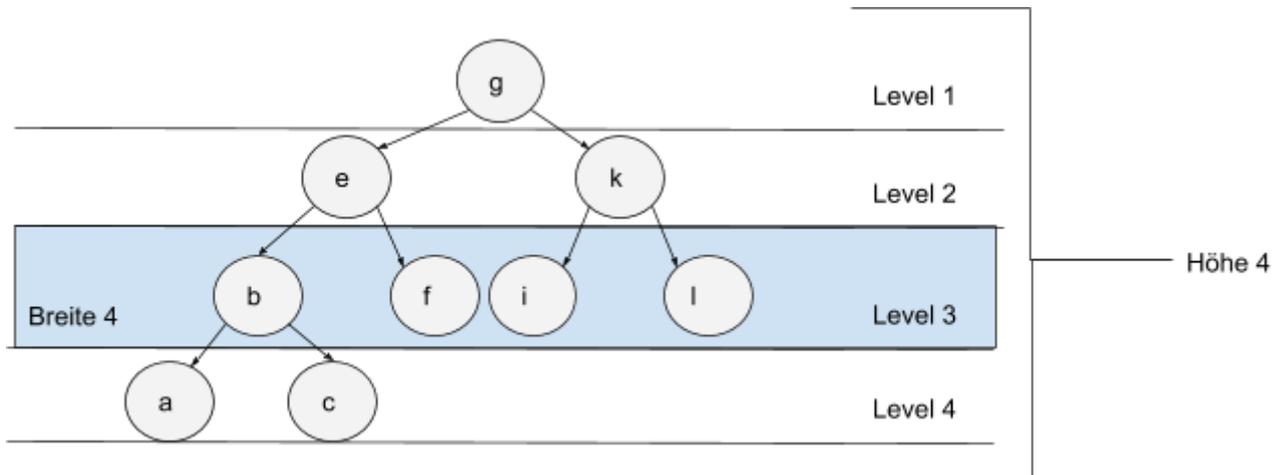
Aufgabe 1: Binärer Suchbaum für sortierte Menge von Zeichen (24 Punkte)

Implementieren Sie eine sortierte Menge für Zeichen (char) in der Klasse *BinaryTreeSet*. Die Schnittstelle ist durch die abstrakte Klasse *Set* gegeben. (Details JavaDoc des Vorgabe Projekts.)

Implementieren Sie die mit **TODO** markierten Funktionen in der Skeleton-Klassen *BinaryTreeSet.java*.

Die Methoden **add**, **contains**, **remove** und **size** sind wie in der Vorlesung besprochen zu implementieren. Des Weiteren enthält die Klasse *BinaryTreeSet* die Methoden **height()**, **width()** und **treeString()** die nicht in der Schnittstelle der *Set* Klasse definiert sind. Diese Methoden modellieren spezielle Binärbaum Operationen.

Die Methode **height()** berechnet die Höhe (siehe Vorlesung) des Binärbaums. Die Methode **width()** berechnet die maximale Breite aller **Level** im Binärbaum. Die Method **treeString()** gibt eine String Repräsentation des Binärbaums zurück.

Beispiel Baum:

Die Method **treeString()** baut mit Hilfe eines *StringBuilders* (JDK Klasse) eine einfache String Repräsentation des Binärbaums auf, die auf der Kommandozeile ausgegeben werden kann. Für obigen Binärbaum würde **treeString()** folgendes Ergebnis liefern.

```

=> g
  => e
    => b
      => a
      => c
    => f
  => k
    => i
    => l
  
```

Implementierungshinweise:

- Verwenden Sie das Vorgabeprojekt **PI2_UE03.zip**.
- Fügen Sie Ihre Implementierung in den mit **TODO** markierten Teilen der Klasse *BinaryTreeSet* ein.
- Ändern sie **keine public Interfaces** vorgegebener Skeleton Klassen (mit Ausnahme der Sichtbarkeiten).
- Halten Sie sich an die **Codierungsrichtlinien** auf der Kurs Website.
- Beachten Sie bei **remove()** die Implementierung die in der Vorlesung besprochen wurde. (Der nächstgrößere Knoten rückt nach.)
- In der Klasse *BinaryTreeSet* können sie gegebenenfalls die Klasse *LinkedListQueue* aus dem Vorgabe Projekt verwenden um Operationen auf dem Baum zu modellieren (wenn es Sinn macht).
- Für das Testen der Klasse *BinaryTreeSet* können sie auf die Klasse *InOrderTreeIterator* zurückgreifen, der die Elemente des Binärbaums in In-Order Traversierungs-Reihenfolge zurück gibt.

Abzugeben ist: Projekt Archiv, Ausgabe Printing des Beispiel Baums (TreeSetMain.java).