**Fortgeschrittene Programmiertechnik**
**Angewandte Informatik**
**SS 2015**
*Prof. Dr. Oliver Bittel*

HOCHSCHULE KONSTANZ
TECHNIK, WIRTSCHAFT UND GESTALTUNG

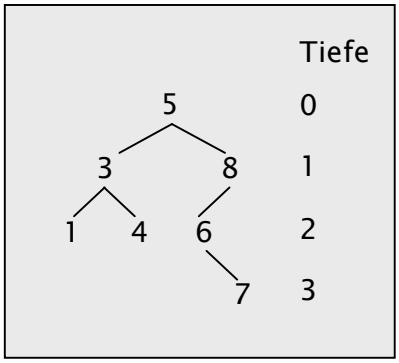
# Aufgabenblatt 10 – Binärer Suchbaum

Auf der Web-Seite finden Sie die Java-Klasse BinarySearchTree.java, die einen binären Suchbaum realisiert.

Die Klasse soll um verschiedene Methoden ergänzt werden.

## main

Schreiben Sie ein Hauptprogramm, das folgenden binären Suchbaum aufbaut.




## public void prettyPrint()

Diese Methode gibt einen binären Suchbaum auf der Konsole aus. Beispielsweise soll die Ausgabe für den oben gezeigten Baum wie folgt aussehen:

```
5
|__3
   |__1
   |__4
|__8
   |__6
      |__#
      |__7
   |__#
```

# ist ein
leerer Teilbaum.

## public void statistik()

Diese Methode ermittelt folgende Informationen über den Binärbaum und gibt diese auf der Konsole aus:

- die Anzahl der Einträge im Baum,
- die Höhe des Baums und

- die durchschnittliche Tiefe aller Einträge.
  Beispielsweise hat der in der Abbildung gezeigte Binärbaum 7 Einträge, eine Höhe von 3 und eine durchschnittliche Tiefe von (0 + 2*1 + 3*2 + 1*3)/7 = 11/7 = 1.57.

## public void subSet(int a, int b, List<Integer> list)

Diese Methode speichert alle Elemente des Suchbaums, die in [a, b] liegen, in der Liste list ab. Achten Sie auf die Effizienz der Methode, indem Teilbäume die keine Elemente aus [a, b] enthalten können, nicht durchsucht werden.

## private boolean isBinarySearchTree(Node p)

Diese Methode prüft, ob der Baum p die Suchbaumeigenschaft erfüllt.

Schreiben Sie isBinarySearchTree(p) als rekursive Methode und verwenden Sie folgende Überlegung:

> Ein binärer Baum p ist genau dann ein korrekter Suchbaum, falls p leer ist oder p.left und p.right korrekte Suchbäume sind und Maximum(p.left) < p.data < Minimum(p.right) gilt.

Folgende Hilfsmethoden könnten hilfreich sein:

- Node minimum(Node p):
  liefert den kleinsten Knoten des Baums p zurück. Rückgabe ist null, falls p = null ist.
- Node maximum(Node p):
  liefert den größten Knoten des Baums p zurück. Rückgabe ist null, falls p = null ist.
- boolean lessThan(Node p, Node q):
  liefert true zurück, falls p oder q = null ist. Falls p != null und q != null ist, wird das Ergebnis des Vergleichs „p.data < q.data“ zurückgeliefert

## Testen von isBinarySearchTree

Diese Methode wird sinnvollerweise als Nachbedingung verwendet, um z.B. zu prüfen, ob die insert bzw. remove-Methode korrekt implementiert ist (siehe auch 2-23 bis 2-30 im Skript).

Schreiben Sie in die letzte Zeile von insert und remove:

```
assert isBinarySearchTree(root);
```

Aktivieren Sie assert durch die VM-Option „-ea“. Bauen Sie nun absichtlich in insertR einen Fehler ein, in dem Sie z.B. „left“ mit „right“ vertauschen. Jetzt müsste die Nachbedingung fehlschlagen und einen „Assertion Error“ auslösen. Korrigieren Sie jetzt wieder Ihren Fehler. Die Nachbedingung müsste jetzt wieder erfüllt sein (beachten Sie, dass natürlich die Methode isBinarySearchTree korrekt implementiert sein muss).

## Abgabe

Führen Sie Ihr Programm vor.