Übungsserie 3: Lösungen

Aufgabe 1: Verständisfragen zu Binärbäumen

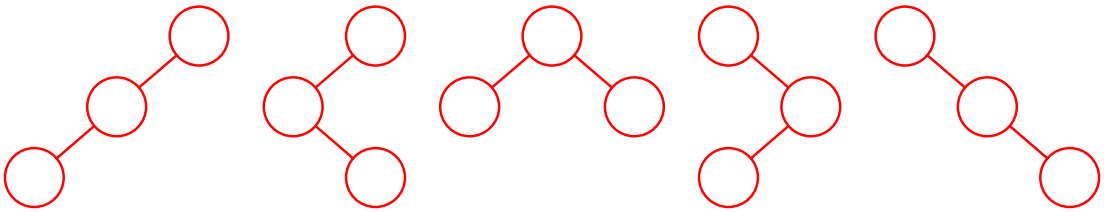
a) Wie viele verschiedene Binärbäume kann man aus den drei Knoten ***A,B*** und ***C*** bilden?

b) Liefert ein ***Preorder***-, ***Postorder***- oder ***Inorder***-*Traversierung* eines Binärbaums die *Blätter (= externe Knoten)* in gleicher oder verschiedener Reihenfolge?

c) Es soll gezeigt werden, dass ein binärer Baum mit ***n*** *Knoten* genau ***n-1*** *Kanten* enthält.

Lösung:

a) Es gibt folgende 5 Grundformen plus alle Varianten mit vertauschten A, B, C:



Insgesamt ergeben sich somit 5 * Anzahl Permutationen von A, B, C also 5 * 3! = 30 Bäume.
Wichtig: je nach Einfügen und Manipulier-Reihenfolge ergeben sich völlig andere Bäume!

b) Die Reihenfolge der Blätter ist tatsächlich immer die gleiche!
Z.B. bei obigem Baum in der Mitte (mit ***A*** in der Root):
Die Reihenfolge der beide Blätter ***B*** und ***C*** ist immer zuerst ***B*** und dann ***C***.

c) Diese Aufgabe lässt sich mithilfe der vollständigen Induktion lösen:
$n = 1$
Ein Baum mit nur einem Knoten hat keine Kanten.
$\surd$

$n \rightarrow n+1$
Wenn man einen Knoten hinzufügt, so muss auch genau eine Kante hinzugefügt werden.
Somit hat der Baum stets eine Kante weniger als Knoten.

Aufgabe 2: Binärbaum-Implementation mit Array

Es soll ein binärer Baum mit der Klasse `VectorTree` implementiert werden, welche den Baum mit einer Array-List realisiert.
Ihre Klasse soll u.a. folgende Methoden beinhalten:

- `root()`
- `isRoot(k)`
- `leftChild(k)`
- `rightChild(k)`
- `parent(k)`
- `isInternal(k)`
- `isExternal(k)`

Hinweise:

- Mehrfache (gleiche) Keys werden nicht unterstützt.
- Die Länge der Array-List (Java: *`ArrayList`*; Python: *`list`*) ist immer eine 2er-Potenz. Die Array-List muss nur vergrössert werden wenn nötig, nicht verkleinert.

Lösung:

siehe: ***uebung03/ml/aufgabe02/VectorTree.java***
resp: ***uebung03/ml/aufgabe02/vector_tree.py***