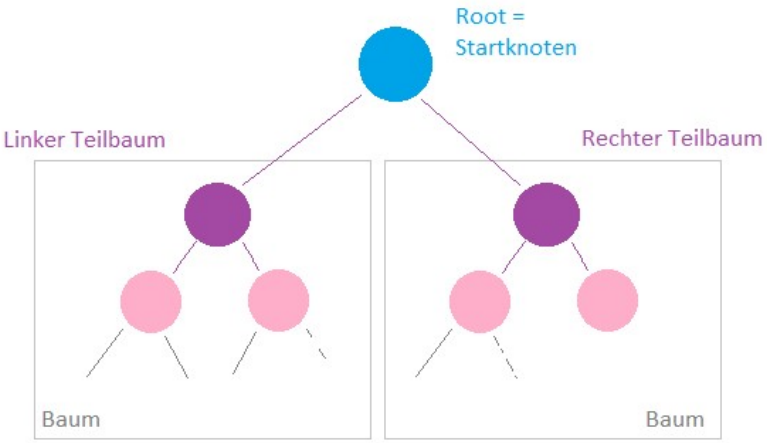
[A04] Baum traversieren

Grafische Beschreibung



Textuelle Beschreibung

Bei diesem Baum handelt es sich um einen Binärbaum, hierbei kann das Root Element, welches dem Startpunkt entspricht, bis zu 2 Kinder (= Teilbäume) haben. Nähere Informationen zum Binärbaum sind auch in A05 zu finden.

Der Algorithmus für „getWordsWithPrefix“ arbeitet rekursiv mit einer Liste an Elementen, wo nach und nach die Kinder von links nach rechts durchlaufen werden und Element für Element hinzugefügt werden. Die Methode wird abgebrochen, sobald kein weiteres Kind mehr gefunden werden kann.

Beim „countWordsInSubTree“ wird der gesamte Baum rekursiv durchlaufen und der Counter um 1 erhöht, solange Kinder existieren.

Laufzeit

- countWordsInSubTree(Wort w) -> O(log(n))
- getWordsWithPrefix(String prefix) -> O(log(n))
- getWordsWithPrefix(String prefix, Wort wort) -> O(log(n))

Wichtige Elemente

```
public int countWordsInSubTree(Wort w) {

   if (w == null)  // terminate recursion
      return 0;

   return countWordsInSubTree(w.getLeft()) + countWordsInSubTree(w.getRight()) + 1;
}


public Set<String> getWordsWithPrefix(String prefix) {
   return getWordsWithPrefix(prefix, root);
}


public Set<String> getWordsWithPrefix(String prefix, Wort wort) { // overloading method with
additional parameter -> otherwise recursive call not possible
   Set<String> set = new HashSet<>();

   if (wort == null)  // check if word "exists" if not return set and stop any further
recursive calls
```

```
      return set;

   if (wort.getWort().startsWith(prefix)) // check if word has prefix
      set.add(wort.getWort());         // if it has add it to the list

                                                // build set through recursive call
   set.addAll(getWordsWithPrefix(prefix, wort.getLeft()));       // addAll so the return value
of type set can be added // recursive call based on left child
   set.addAll(getWordsWithPrefix(prefix, wort.getRight()));   // addAll so the return value of
type set can be added // recursive call based on right child

   return set;
}
```