ADP Aufgabe 1, Abgabe 1

Team 1
Hugo Protsch, Justin Hoffmann

November 11, 2020

1 Formales

1.1 Aufgabenaufteilung

Die Entwürfe wurden zusammen entwickelt.

1.2 Quellenangaben

Es wurden lediglich Vorlesungsmaterialien verwendet.

1.3 Bearbeitungszeitraum

Für die Bearbeitung haben wir in etwa 8 bis 10 Stunden benötigt.

1.4 Aktueller Stand

Die Entwürfe sind fertiggestellt und müssen implementiert werden.

1.5 Änderungen des Entwurfes

– Nicht zutreffend –

2 Entwürfe

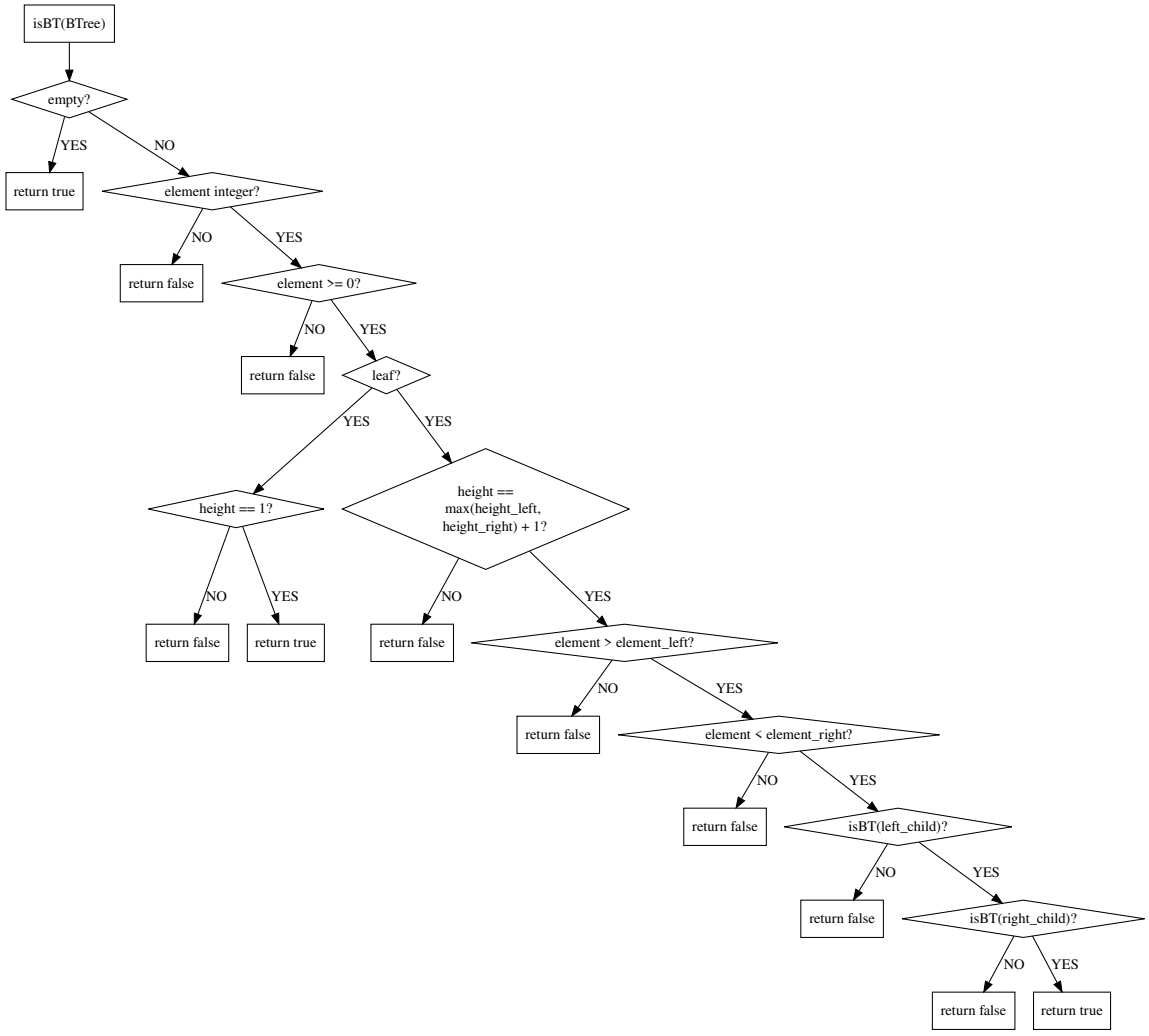
2.1 InitBT

Liefert einen leeren Baum. Der Entwurf ist trivial. Die erwartete Laufzeit beträgt $\Theta(1)$

2.2 IsEmptyBT

Prüft einen Baum auf Leerheit. Da der Baum nicht auf Korrektheit zu prüfen ist, ist der Entwurf trivial. Die erwartete Laufzeit beträgt $\Theta(1)$

2.3 IsBt



Prüft die vorgegebenen Rahmenbedingungen eines übergebenen Baums. Hierbei werden alle Wahrheitswerte der verschiedenen Vorgaben von jedem Knoten miteinander in einen logischen Zusammenhang gebracht.

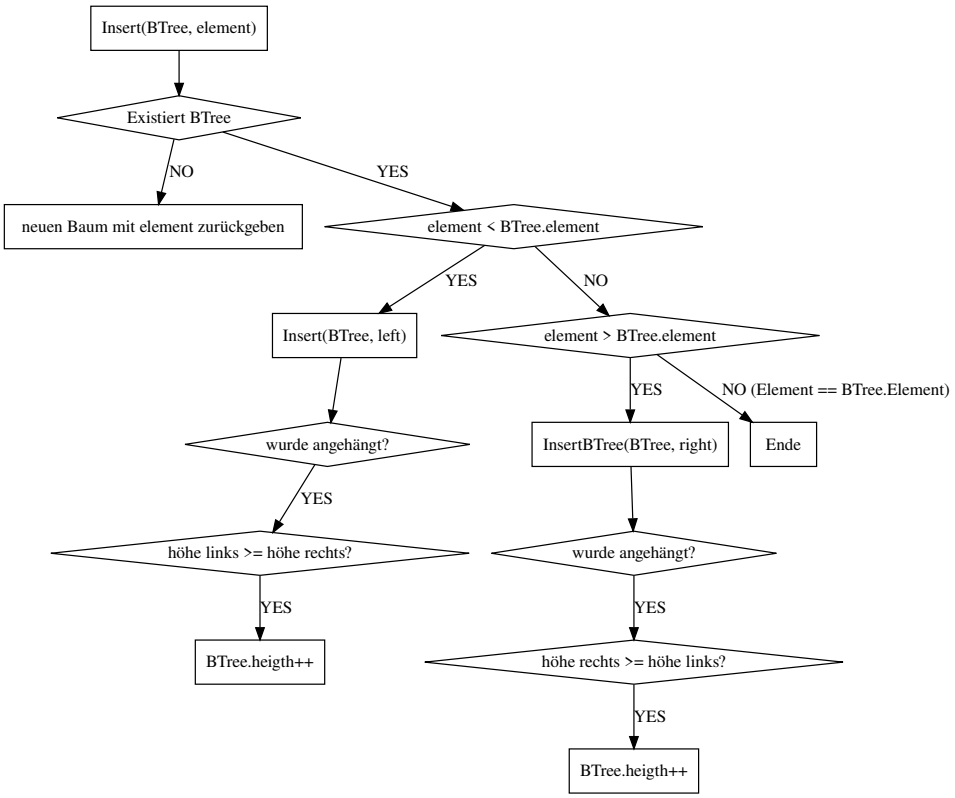
Die erwartete Laufzeit beträgt $O(n)$

2.4 EqualsBT

Prüft auf semantische Gleichheit zwischen zwei Bäumen. Entwurf ist ebenfalls trivial, da die inOrderBT-Funktion auf beide Bäume angewendet wird und die resultierenden Listen verglichen werden.

Die erwartete Laufzeit beträgt $O(n)$, da die Bäume unabhängig voneinander durchlaufen werden, im Anschluss wird ein Mal über eine Liste von n Elementen iteriert.

2.5 InsertBT

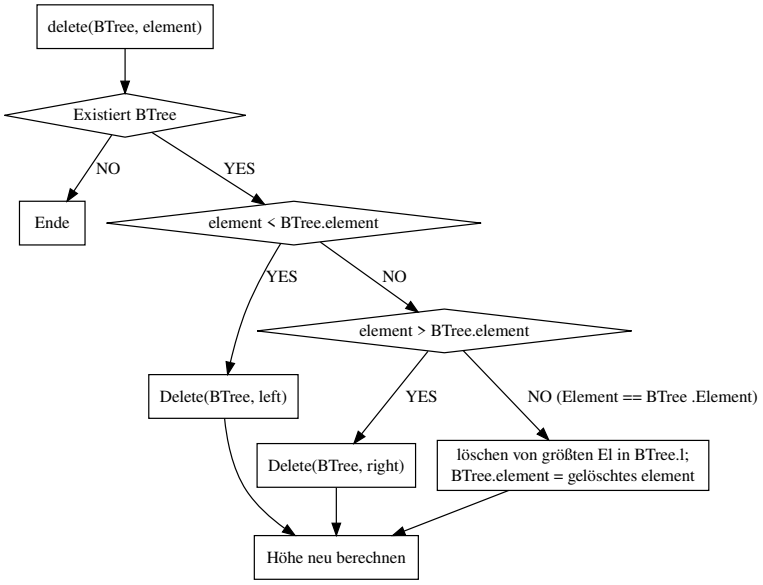


Fügt ein übergebenes Element einem übergebenem Baum hinzu. Zum Traversieren des Baumes wird bei jedem Schritt überprüft, ob das einzufügende Element kleiner oder größer dem derzeitigen Knoten-Element ist. Ist es kleiner, geht man nach links, größer nach rechts. Wenn bereits ein Knoten

mit dem gleichen Wert existiert, wird der Baum unverändert zurückgegeben. Wurde ein leerer Platz gefunden, wird das Element dort eingefügt. Falls ein Element eingefügt wurde, muss die Höhe aktualisiert werden. Diese wird inkrementiert, falls der Zweig, der an ein Element angefügt wurde, höhenbestimmend ist.

Die erwartete Laufzeit beträgt $O(n)$ bzw. $\Theta(log(n))$, da der Baum im Worst-Case nicht ausgeglichen ist. Das hat zur Folge, dass alle Elemente einmal durchlaufen werden müssen. Durchschnittlich wird der Baum mit einer logarithmischen Komplexität traversiert, da bei einem ausgeglichenem Baum durch die vertikale Traversierung nur $log(n)$ Elemente durchlaufen werden.

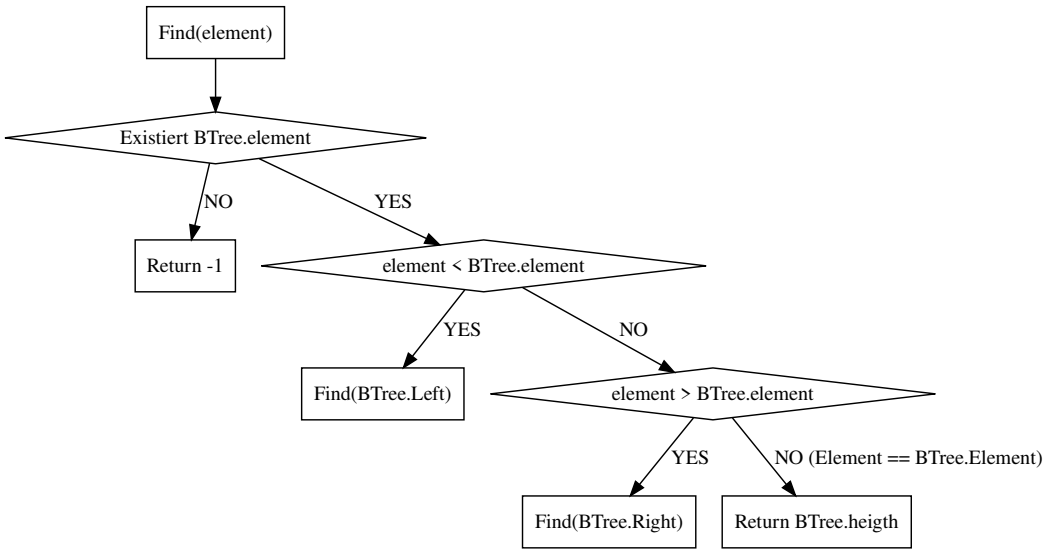
2.6 DeleteBT



Löscht ein übergebenes Element aus einem übergebenem Baum. Die Traversierung des Baums ist gleich wie bei InsertBT. Ist das Ende des Baumes erreicht, ohne das Element gefunden zu haben, wird der Baum unverändert zurückgegeben. Wird das Element aufgefunden, kommt an seine Stelle das Element aus dem linken Teilbaum mit dem höchsten Wert. Die Höhe wird Bottom-Up angepasst.

Die erwartete Laufzeit beträgt $O(n)$ bzw. $\Theta(log(n))$ aufgrund der bei InsertBT erwähnten Umstände.

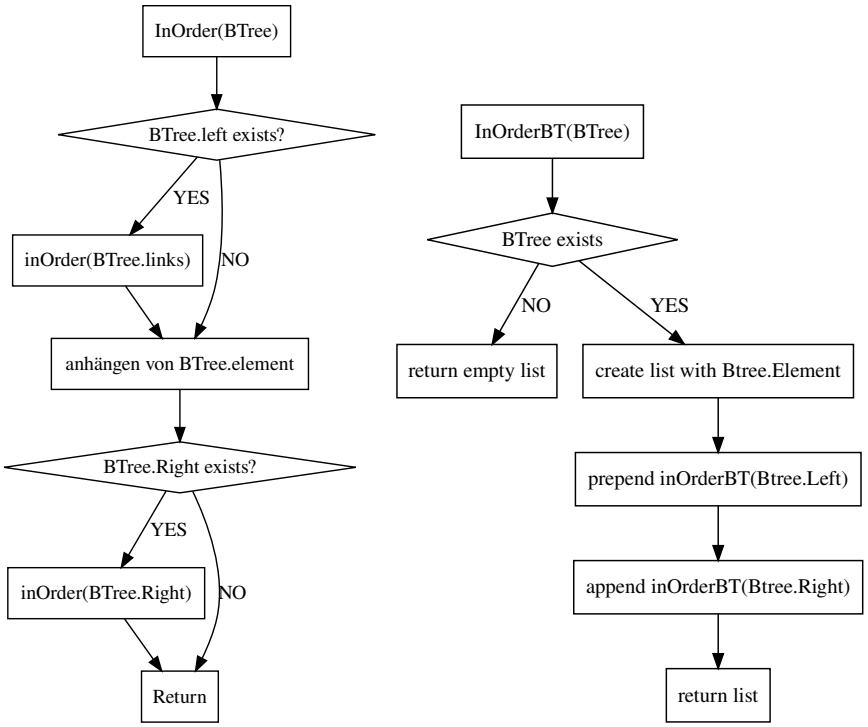
2.7 Find



Gibt die Höhe eines übergebenem Elements in einem übergebenem Baum aus. Traversieren des Baums funktioniert hierbei gleich wie in InsertBT. Ist das Element gefunden, wird die Höhe des Knotens ausgegeben.

Die erwartete Laufzeit beträgt $O(n)$ bzw. $\Theta(log(n))$ aufgrund der bei InsertBT erwähnten Umstände.

2.8 InOrderBT



Gibt den Baum in Inorder in einer Liste aus. Es wird eine Liste mit dem Element des Knotens erstellt, wobei die Inorder des linken Kindes vorangestellt und die Inorder des rechten Kindes herangehängt wird. Ist das Ende des Baumes erreicht, wird eine leere Liste zurückgegeben.

Hier haben wir zwei Optionen dargestellt: Zum Einen kann bei jedem Funktionsdurchlauf der derzeitige Knoten selbst betrachtet werden (rechts). Zum anderen gibt es die Möglichkeit, vorausschauend zu arbeiten und immer auf die Kinder des derzeitigen Knotens achten (links). Die Auswahl des Entwurfes ist implementationsabhängig.

Die erwartete Laufzeit beträgt $\Theta(n)$, da jedes Element des Baums durchlaufen wird. Hierbei ist zu beachten, dass die Gesamtkomplexität des Weiteren von der Komplexität der Append- bzw. Prepend-Operation der Liste abhängt. Falls die Komplexität dieser Operation $O(n)$ beträgt, ergibt sich daraus eine Gesamtkomplexität von $O(log(n) \cdot n)$

3 Laufzeitmessung

Für den Versuch der Laufzeitmessung werden jeweils die Funktionen mit variierender Eingangsgröße getestet. Die Laufzeit wird anschließend mit besagter Eingangsgröße in Relation gesetzt. Wir vergleichen den gemessenen funktionellen Zusammenhang mit dem Erwarteten, welcher in den Entwürfen der jeweiligen Funktion beschrieben ist.

Die Rahmenbedingungen der Messung sind hierbei vorgegeben.