

# **Beleg 3**

## Binary Tree

### **Benutzerhandbuch**



Student	Hahmann Felix (544657), Petsche Sven (545821)
Studiengang	Angewandte Informatik
Semester	2. Semester
Modul	Programmierung 2 (Java)
Dozent	Herr Prof. Dr. R. Kosciolowicz
Abgabetermin	23.07.2014

## Inhaltsverzeichnis

1.	Aufgabenstellung .....	3
2.	Generelles .....	4
3.	Bedienung .....	5
4.	Testbeispiele .....	9
5.	Quellen.....	10

# 1. Aufgabenstellung

**Programmierung 2**

**B2AI / SS 2014**

**Prof. Dr. R. Kosciolowicz**

**3. Belegarbeit**

**Beginn: 18.6.14**

**Ende: 23.7.14**

Entwickeln Sie ein Testprogramm und eine Klasse zur Verwaltung von sortierten Binärbäumen, die kurze Strings (max. 3 Zeichen) speichern können. Es sollen folgende Klassenmethoden realisiert werden:

1. leeren Baum erzeugen im Konstruktor
2. Knoten hinzufügen
3. Baum anzeigen (alphanumerisch als Werteliste oder grafisch im Dialogfenster)
4. Tiefe des Baumes ausgeben
5. Knoten löschen
6. Baum in externe Datei speichern (ASCII oder binär)
7. Baum aus externer Datei lesen (ASCII oder binär)

Schreiben Sie ein Konsolenprogramm und/oder eine Dialoganwendung um einen Binärbaum interaktiv einzugeben, anzuzeigen, zu verändern und in einer externen Datei abzuspeichern.

Zur Verwaltung des Baumes sind dynamische Speicherobjekte zu verwenden (siehe Vorlesung/Übung). Ein Knoten des Baumes soll mindestens folgende Elemente enthalten (Knotenwert, Nachfolgerknoten links, Nachfolgerknoten rechts). Zusätzlich können folgende Informationen sinnvoll sein (Grad des Teilbaumes, X-Position, Y-Position des Knotens für grafische Anzeige).

Entwerfen Sie zwei Testbeispiele für sortierte Binärbäume mit mindestens 10-14 Knoten. Simulieren Sie folgende Veränderungen und zeichnen Sie das Endergebnis auf Papier (Entfernen von mindestens zwei Knoten und Hinzufügen von mindestens vier neuen Knoten).

Die Kurzdokumentation soll die Testbeispiele als Skizze, eine Bildschirmkopie wichtiger Ausgaben des Programms (Menü, Masken, Baumanzeige) sowie die wichtigsten aufgetretenen Probleme enthalten.

Konsolenlösung

**13 Pkt**

Fakultative Erweiterungen:

- a) Entwickeln Sie eine grafische Oberfläche für die Baumoperationen mit Dialogfeldern und einer zeichnerischen Darstellung des Baumes **9 Pkt**
- b) Ausbalancieren des Baumes als AVL-Baum in einer Methode oder Erzeugung eines fast vollständig ausgeglichenen Baumes in einer Methode **3 Pkt**

Bewertet werden der Programmquelltext 50%, die korrekte Arbeitsweise 20%, die Dokumentation 20% sowie die vorbereiteten Testbeispiele 10%

**Anmerkung:** In Absprache mit Hr. Prof. Dr. Kosciolowicz wird die fakultative Erweiterung b) (AVL-Baum) durch folgende Funktionen ersetzt: Infocfeld mit Information zur letzten Aktion, Logbuch der letzten Aktionen, Löschen des Baumes vom Datenträger mit Auswahlliste, Leeren der Ansicht, dynamische Anpassung des Fensters an die Größe des Binärbaumes (vertikal wie horizontal), autom. Speichern des Baumes beim Beenden in „save.tre“.

## 2. Generelles

Das Programm ermöglicht das Anlegen von binären Suchbäumen. Ein binärer Suchbaum ist eine spezielle Form eines Graphen, der als gewurzelter Baum an jedem Knoten höchstens zwei Kindknoten besitzt. Die Kindknoten lassen sich dabei eindeutig in linkes und rechtes Kind einteilen.

Das oberste Element, das keine Eltern besitzt, ist die Wurzel. Elemente am „Ende“ des Baumes, die keine Kinder besitzen, werden als Blätter bezeichnet. Alle Knoten dazwischen werden als innere Knoten bezeichnet. Bei dem binären Suchbaum können Knoten hinzugefügt und entfernt werden.

Desweiteren können die erstellten binäre Suchbäume auf dem Datenträger gespeichert, vom Datenträger geladen sowie vom Datenträger gelöscht werden. Außerdem ist es möglich, den angezeigten Baum bzw. die Ansicht zu leeren / zu löschen. Somit ist es auch möglich, die gespeicherten Bäume in Form der Dateien zwischen Rechnern und/oder Personen auszutauschen.

Beim Beenden des Programms wird der aktuelle Baum in der Datei `save.tre` gespeichert und kann über die Funktion „Baum laden“ und der Datei `save.tre` wieder hergestellt werden.

### 3. Bedienung

#### Knoten Hinzufügen

Zum Hinzufügen eines Knotens muss ein max. 3-stelliger String in das rot markierte Eingabefeld eingetragen und [ENTER] oder der Button „neuer Knoten“ betätigt werden.



Abb. 1: Knoten hinzufügen

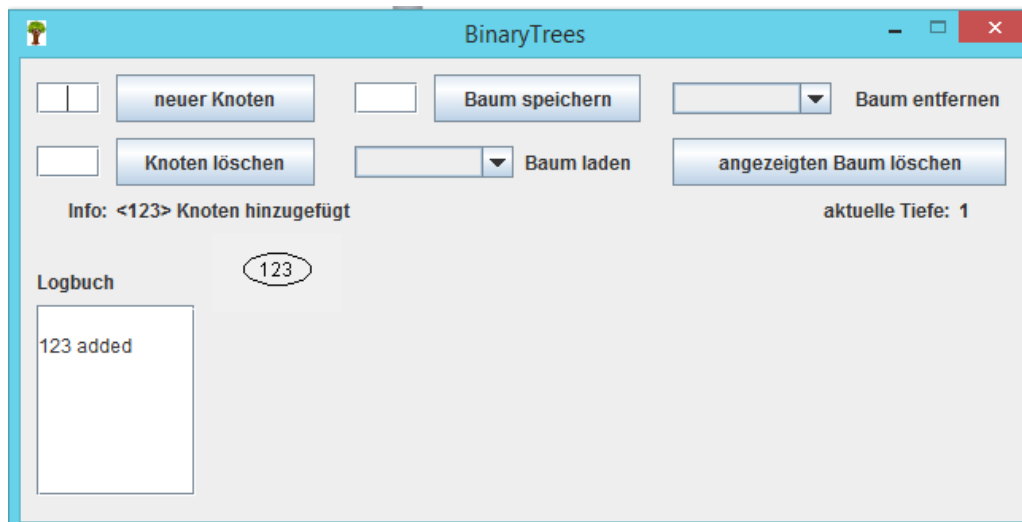


Abb. 2: Knoten hinzugefügt

## Knoten löschen

Zum Löschen eines Knotens muss der Bezeichner/Name des Knotens in das rot markierte Feld eingetragen werden und anschließend [ENTER] oder der Button „Knoten löschen“ betätigt werden. Der Knoten wird dann entfernt und der nächste Eintrag rückt an die Stelle des bisherigen Knotens, wenn der Knoten nicht ein letztes Element (Blatt) war, wie in der darauffolgenden Abbildung ersichtlich.

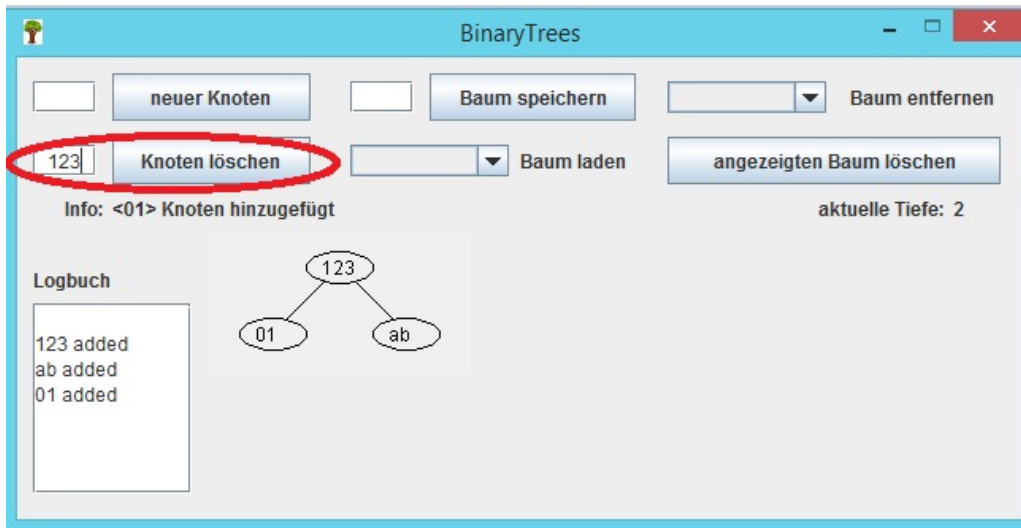


Abb. 3: Knoten löschen

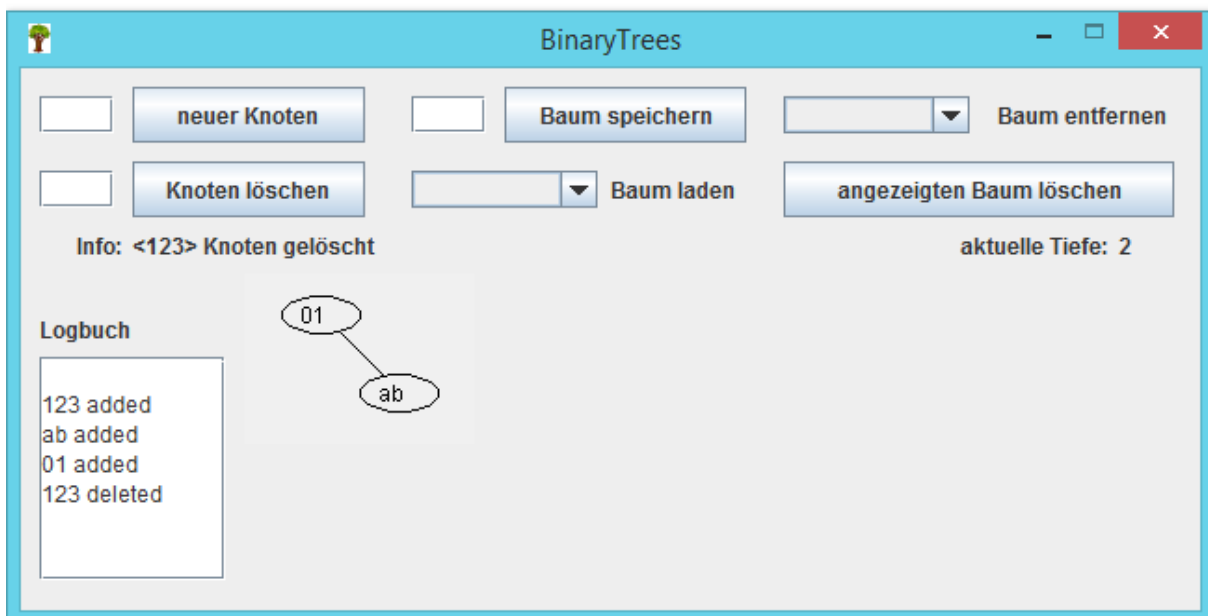


Abb. 4: Knoten gelöscht

## **Baum speichern**

Es ist möglich, den erstellten Baum in einer Datei zu speichern. Somit ist ein Laden des Baumes auch nach Beenden des Programms oder Rechners möglich. Dazu muss der Bezeichner für den Dateinamen im Feld von „Baum speichern“ eingegeben werden und [ENTER] oder der Button „Baum speichern“ betätigt werden.

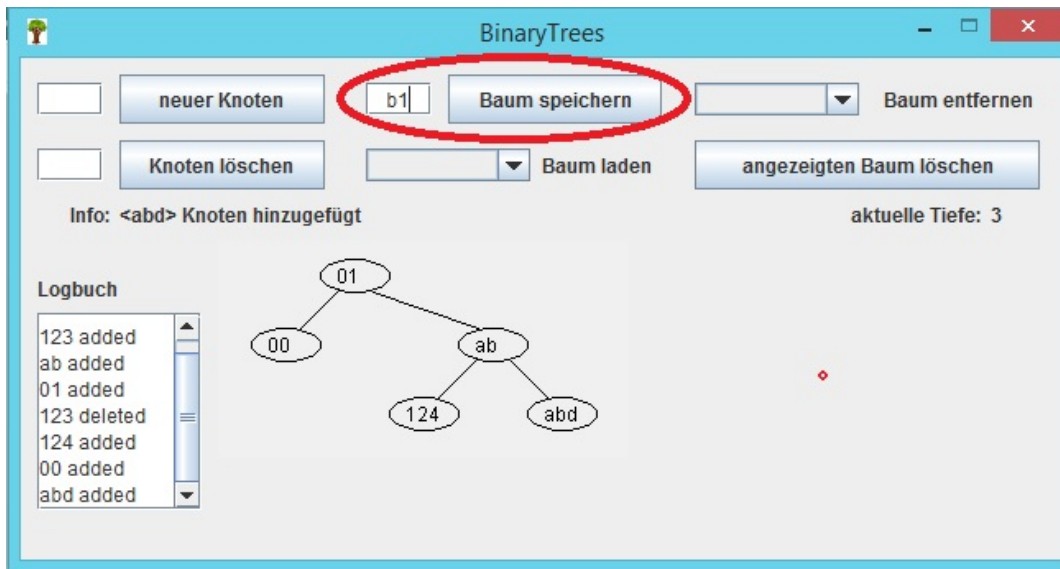


Abb. 5: Baum speichern

## **Baum laden**

Wurden bereits Bäume gespeichert, so kann über die Auswahlliste von „Baum laden“ der gewünschte Baum geladen werden. Das Logbuch wird dabei geleert.

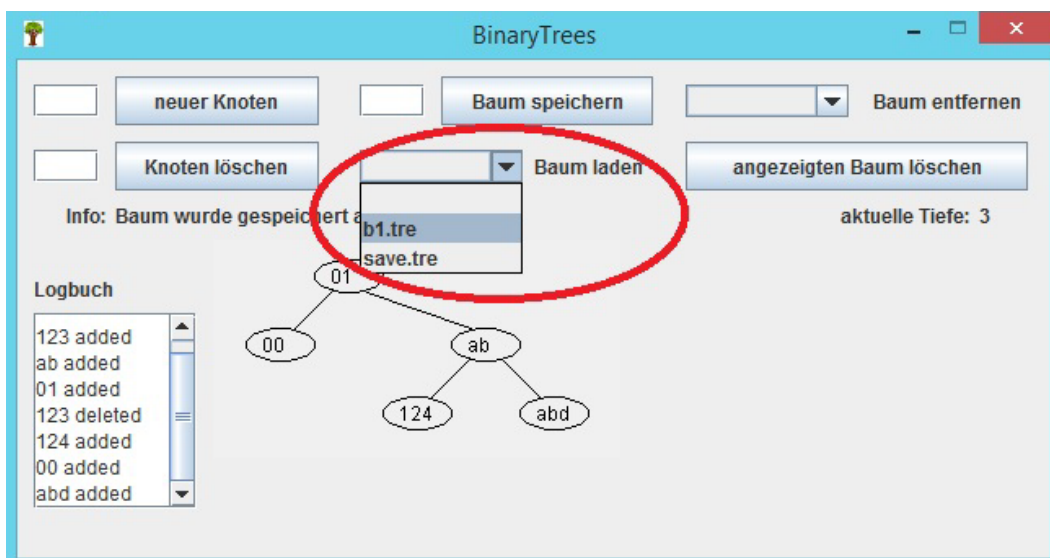


Abb. 6: Baum laden

### **Baum entfernen**

Nach dem gleichen Prinzip kann auch ein gespeicherter Baum entfernt werden. Die Datei wird dabei vom Datenträger gelöscht und der Baum verschwindet aus der Auswahlliste von „Baum laden“ und „Baum entfernen“.

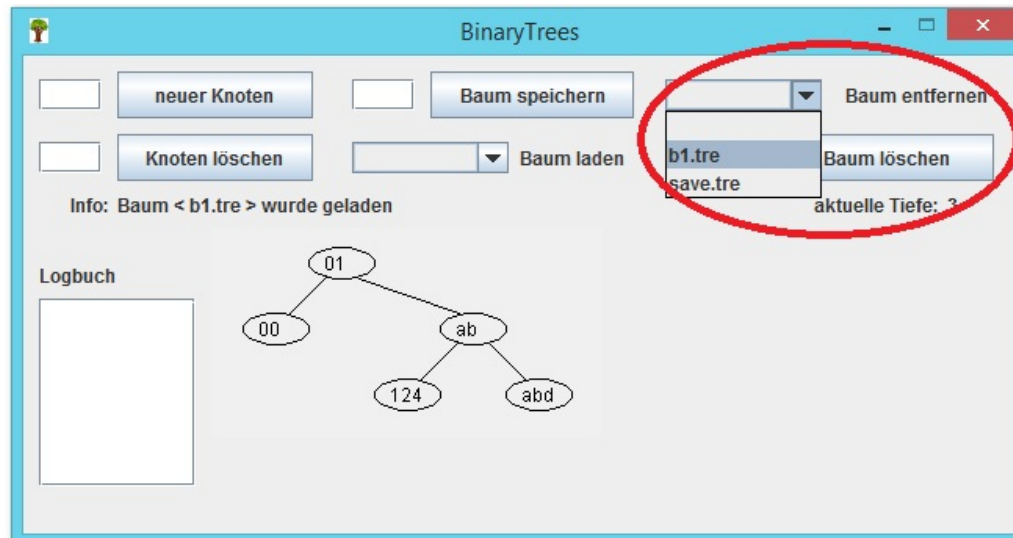


Abb. 7: Baum entfernen

### **Angezeigten Baum löschen**

Mit dem Button „Angezeigten Baum löschen“ wird der dargestellte Baum entfernt. Das Erstellen von Knoten kann jetzt wieder von vorn beginnen.

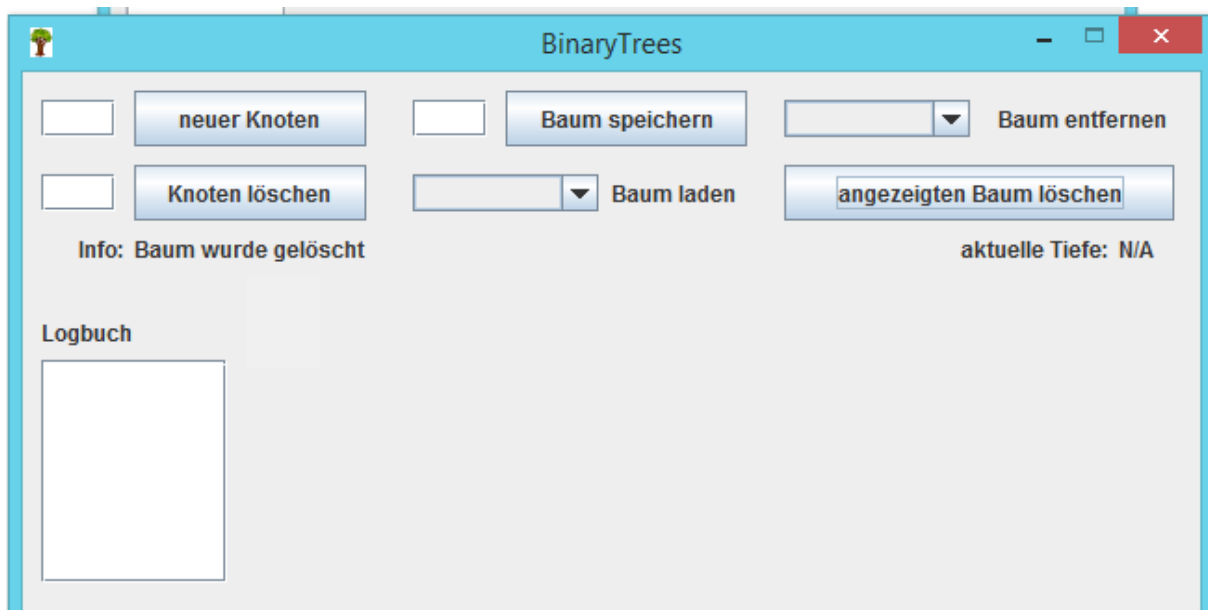


Abb. 8: angezeigten Baum löschen



## 4. Testbeispiele

In der Aufgabenstellung werden zwei Testbeispiele mit 10-14 Knoten und den Entfernen von min. zwei und dem hinzufügen von min. 4 Knoten gefordert.

Nachfolgend Testbeispiel 1 mit 12 Knoten:

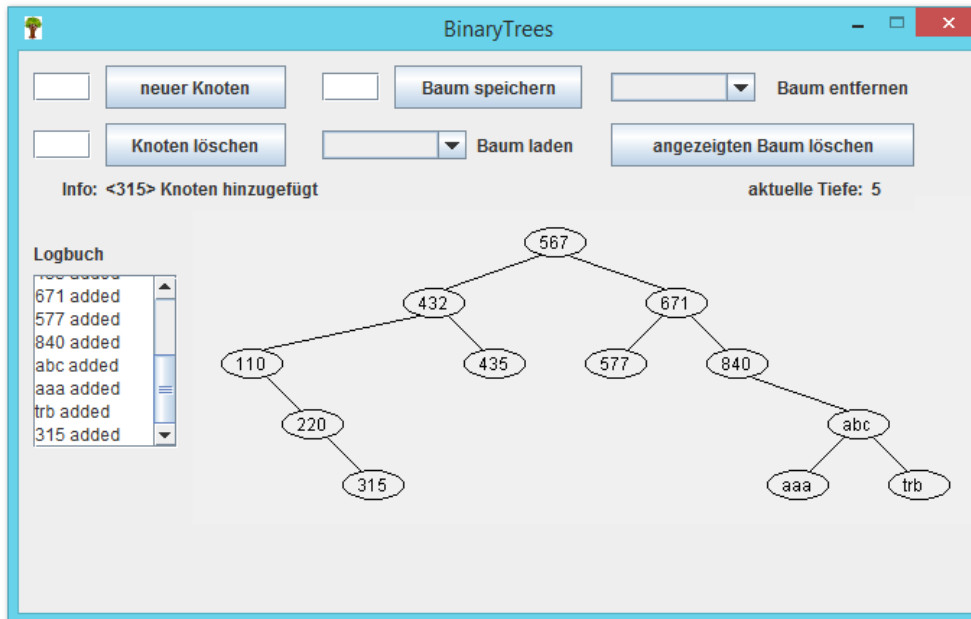


Abb. 9: Testbeispiel 1

Folgende Knoten wurden entfernt (vgl. Logbuch): 840, 110

Folgende Knoten wurden hinzugefügt (vgl. Logbuch): zzz, rsa, aab, 564

Testbeispiel 2 mit den erfolgten Änderungen:

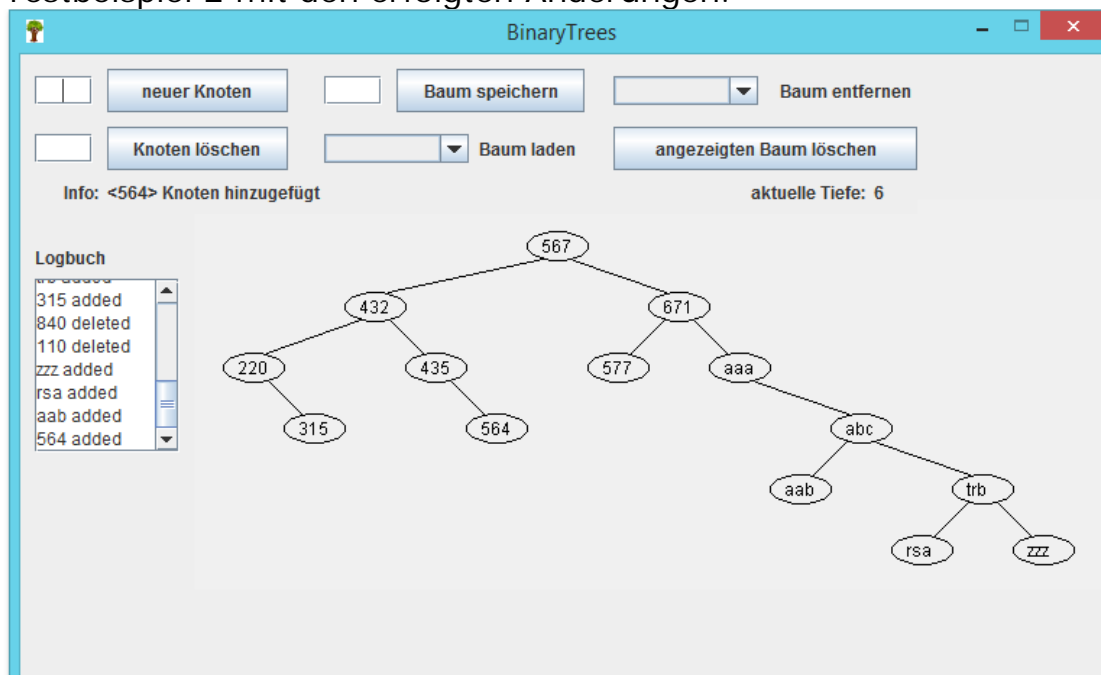


Abb. 10: Testbeispiel 2

## 5. Quellen

[http://stocklogos.com/sites/default/files/styles/logo-medium-alt/public/spintherism-binary-tree\\_alt1.png?itok=Gw3D2qu5](http://stocklogos.com/sites/default/files/styles/logo-medium-alt/public/spintherism-binary-tree_alt1.png?itok=Gw3D2qu5)

<http://de.wikipedia.org/wiki/Bin%C3%A4rbaum>