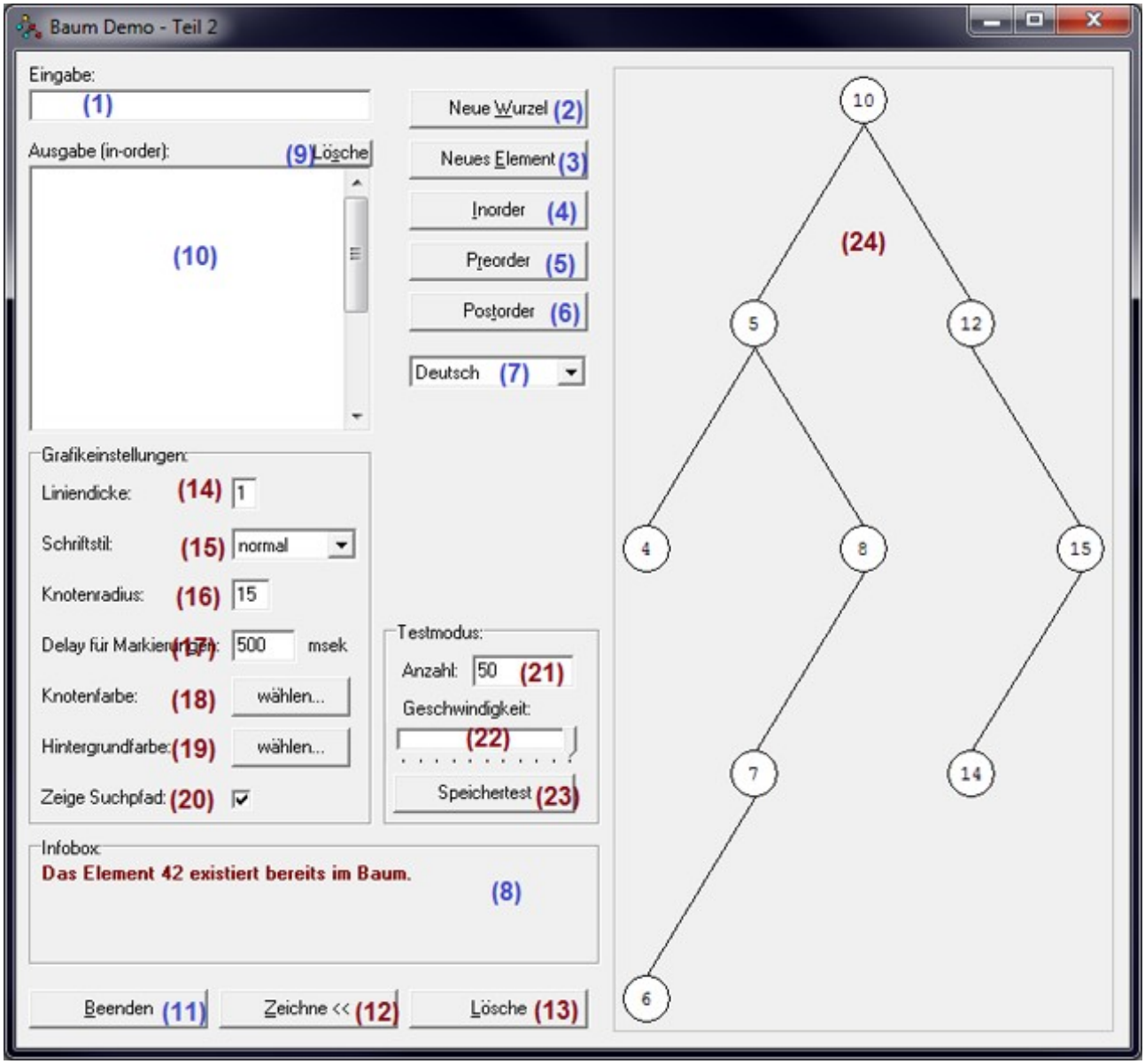
Schulprojekt – Binäre Bäume – Teil 2

Funktionsumfang



Teil 1:

(1) Dies ist das Eingabefeld, in dem die Werte neuer Elemente angegeben werden können. Mit der „ENTER“-Taste wird automatisch ein neues Element in den Baum eingefügt. Besteht noch keine Wurzel, so wird diese angelegt. Andernfalls wird lediglich ein weiteres Element erzeugt und hinzugefügt.
Gültige Werte sind lediglich Zahlenwerte im Bereich von MIN_INT_VALUE bis MAX_INT_VALUE

(2) Dieser Button erzeugt eine neue Wurzel. Existiert bereits ein Baum, so wird der Nutzer gefragt, ob der alte Baum gelöscht und ein neuer erstellt werden soll.

(3) Dieser Button erstellt ein neues Element und fügt dieses in den bestehenden Baum ein. Es

muss erst eine Wurzel erstellt werden, bevor die Aktion des Buttons ausgeführt werden kann.

(4) Dieser Button traversiert den bestehenden Baum in-order und gibt das Ergebnis im Ausgabefeld (10) aus. Die Traversierung kann nur durchgeführt werden, wenn zumindest eine Wurzel vorhanden ist.

(5) Dieser Button traversiert den bestehenden Baum pre-order und gibt das Ergebnis im Ausgabefeld (10) aus. Die Traversierung kann nur durchgeführt werden, wenn zumindest eine Wurzel vorhanden ist.

(6) Dieser Button traversiert den bestehenden Baum post-order und gibt das Ergebnis im Ausgabefeld (10) aus. Die Traversierung kann nur durchgeführt werden, wenn zumindest eine Wurzel vorhanden ist.

(7) Über diese Auswahlbox kann in-time die Sprache des Programms geändert werden.

(8) Die Infobox zeigt in grüner Schrift die letzte Aktion des Benutzers an. Konnte eine Aktion des Benutzers nicht erfolgreich durchgeführt werden oder wenn ein Fehler aufgetreten ist, so wird dies in roter Schrift in diesem Feld dem Nutzer mitgeteilt.

(9) Mit diesem Button kann der Inhalt des Ausgabefeldes (10) gelöscht werden.

(10) Dies ist das Ausgabefeld für die Traversierungen.

(11) Dieser Button löscht den aktuellen Baum und beendet das Programm.

Teil 2:

(12) Über diesen Button kann der Zeichenmodus aktiviert bzw. deaktiviert werden.

(13) Dieser Button entfernt das Element mit dem Wert, der im Eingabefeld eingetragen wurde aus dem Baum.

(14) Stellt die Linien-dicke der Knoten und der Verbindungslinien auf der Zeichenfläche ein.

(15) Stellt den Schriftstil der Knotenbeschriftungen auf der Zeichenfläche ein.

(16) Stellt den Radius der Knoten in Pixel auf der Zeichenfläche ein.

(17) Stellt die Wartezeit in Millisekunden ein, die nach einer farblichen Markierung eines Knotens gewartet werden soll, bevor die Markierung wieder entfernt wird.

(18) Stellt die Hintergrundfarbe der Knoten auf der Zeichenfläche ein.

(19) Stellt die Hintergrundfarbe der Zeichenfläche ein.

(20) Stellt ein, ob der Suchpfad bei jeder Suche im Baum (blau) angezeigt werden soll.

(21) Stellt die Anzahl an Elementen ein, die in den Baum in zufälliger Reihenfolge eingefügt werden sollen.

(22) Aktiviert den random Tester und stellt gleichzeitig die Geschwindigkeit ein.
Steht der Regler ganz rechts, so ist der random Tester deaktiviert.
Steht der Regler ganz links läuft er mit maximaler Geschwinigkeit.
Die ersten beiden Positionen sind spezielle hohe Geschwindigkeiten bei denen eine angegebene Wartezeit (17) keine Auswirkung hat.
Ab der Position drei wird der random Tester pro weitere Position um eine Sekunde langsamer, beginnend bei einer Sekunde.

(23) Dieser Button führt einen Speichertest durch. Dabei wird der Baum auf drei verschiedene Weisen gefüllt und auf mehrere unterschiedliche Weisen wieder geleert / gelöscht. Es werden zu Beginn, beim maximalen Speicherverbrauch und zum Ende eines jeden Teiltests der jeweilige Speicherverbrauch gemessen und in einem Bericht festgehalten, der am Ende angezeigt wird.
HINWEIS: Da nicht sehr viele Elemente in den Baum eingefügt werden, muss im Quellcode folgende Zeile in der Unit unitElement einkommentiert sein:

```
memTest : Array[0..100] of Array[0..100] of Array[0..100] of String;
```

Dies bewirkt, dass pro neuem Element im Baum ca. 4 MB Speicher mehr benötigt werden. So fallen Speicherlecks am schnellsten auf.

(24) Dies ist die Zeichenfläche auf dem der Baum visualisiert wird. Der Visualisierung liegt der Reingold & Tilford Algorithmus zugrunde. (*Hat mich drei ganze Tage gekostet den zu verstehen und zu implementieren! :-) Benutzte Quellen stehen im Code Unit unitTreeViewer...*)