

Erinnert ihr euch noch, wie wir versucht haben bei einem sortierten Array die Suchzeit nach einem bestimmten Wert zu halbieren?

Hier ist unser Array:

3 5 9 15 18 22 38

Genau! Wir haben ihn aufgeteilt.

3 5 9 15 18 22 38

Dann suchen wir entweder rechts oder links weiter. Wie können wir die Zeit weiter verkürzen?

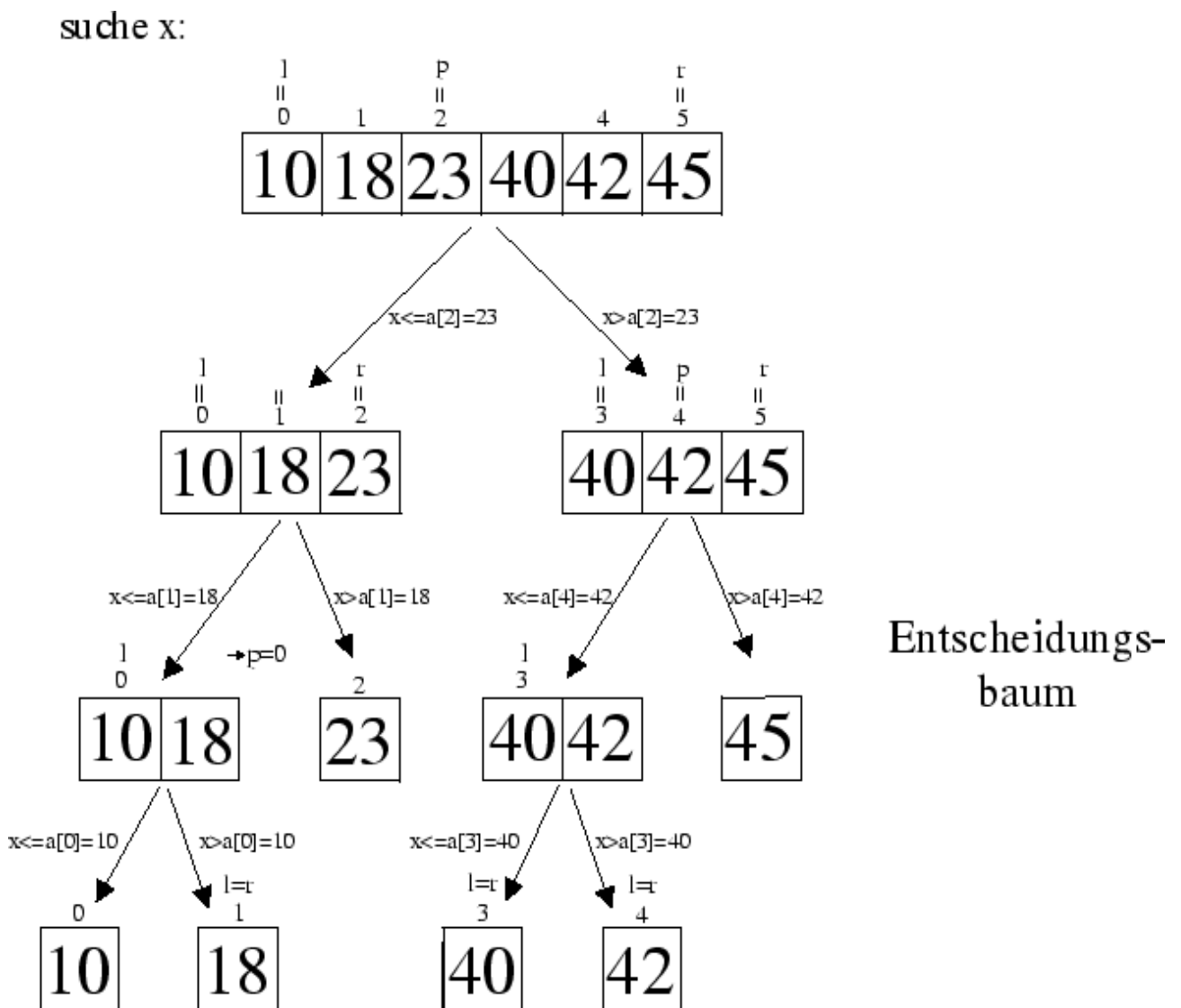
Na ja, so schwer wars nicht oder?

3 5 9 15 18 22 38

Der nächste Schritt ist dann ebenfalls klar: Immer so weiter bis wir entweder die Zahl gefunden haben oder eben nicht.

Das nennt man einen Entscheidungsbaum.

Hier noch ein Beispiel:



Wir sparen eine Menge Zeit, wenn wir so suchen!

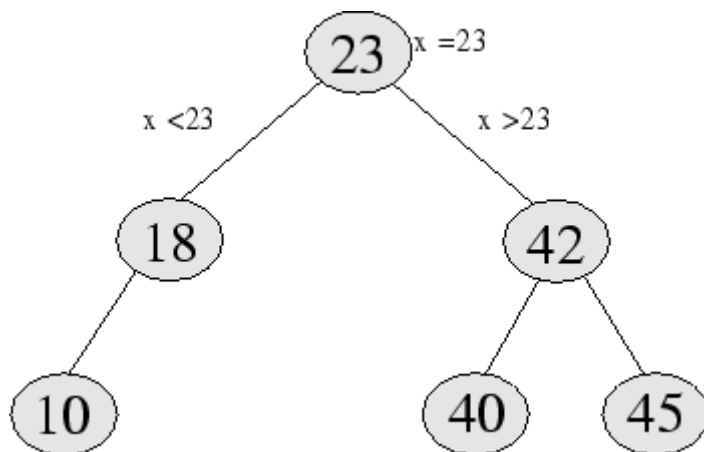
Genau deshalb brauchen wir eine neue Datenstruktur, da der Array zu eingeschränkt ist.

Was müsste ein einzelnes Objekt wissen? Dann könnt ihr entscheiden, wie es in einer so aufgebauten Struktur aussehen könnte.

Schwierig oder?

Eigentlich aber nicht. Ein Knoten (node) besitzt einen Wert und einen Verweis auf seine rechten und linken Teilbäume.

Das Beispiel von oben sieht dann so aus:



Eine Suche sieht dann folgendermaßen aus:

Wenn $x=23$ dann gefunden!

Sonst

wenn $x < 23$ suche links weiter

wenn leer: nicht vorhanden

wenn $x > 23$ suche rechts weiter

wenn leer: nicht vorhanden

Bei diesem Baum gilt:

Der Schlüssel in einem Knoten ist größer als alle Schlüssel im linken TB und kleiner als alle im rechten.

Merken:

Der Inhalt eines Knotens heißt dabei Schlüssel!

Der erste Knoten heißt Wurzel!

Ein Baum mit Knoten, die jeweils nur zwei
Nachfolger haben heißt Binärbaum!

Aufgabe 1):

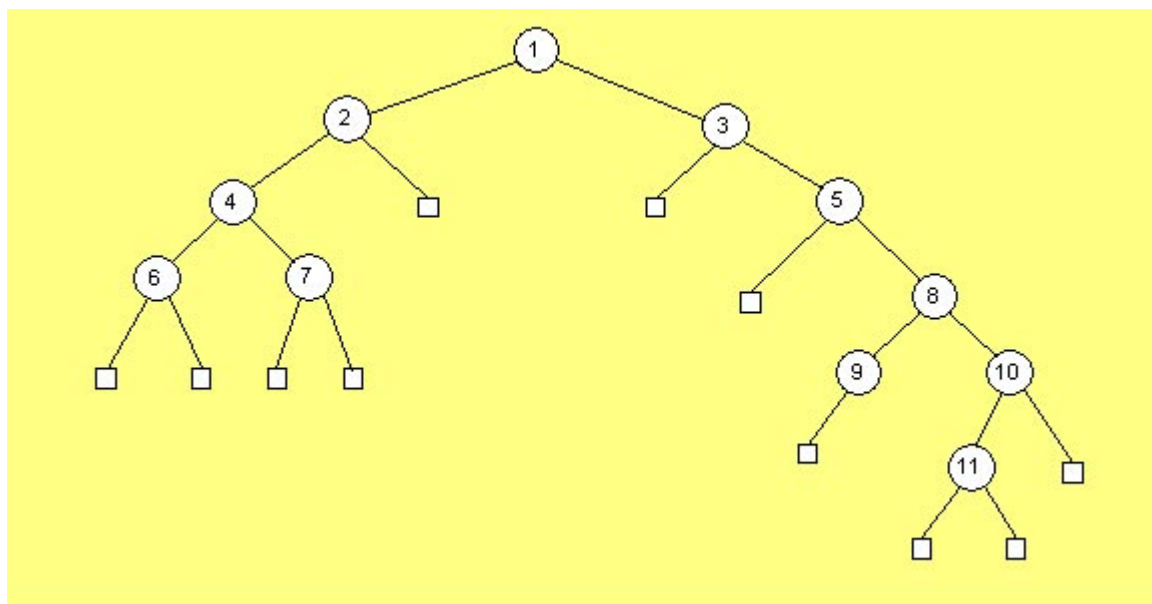
Erstellt einen ausgeglichenen Baum (dh. das der rechte und linke TB die gleiche Höhe ± 1 haben. Höhe ist dabei der Abstand zu Wurzel.) mit den Zahlen von 1-13.

Weiter geht's:

Nochmals zum Binärbaum:

Der binäre Baum ist eine besondere Baumstruktur. Er definiert sich dadurch, dass jeder Knoten (= Element des Baums) genau 2 Nachfolger haben muss. Diese Eigenschaft macht ihn zu einer der am häufigsten verwendeten Baumstrukturen.

Beim binären Baum unterscheidet man zwischen der Wurzel, den inneren und äußeren Knoten. Die Wurzel ist der Grundknoten des Baumes, da sie keinen Vorgänger besitzt. Ist dieser Knoten leer, so ist der gesamte Baum leer. Äußere Knoten nennt man jene, welche keinen Nachfolger haben. Diese Knoten befinden sich zumeist auf der untersten Ebene eines Baumes, sie müssen leer sein, um der Definition nicht zu widersprechen. Äußere Knoten werden oft nicht rund, sondern eckig dargestellt, damit man sie von den inneren Knoten unterscheiden kann. Alle anderen Knoten, denen ein Wert zugewiesen ist, heißen innere Knoten. Jeder von diesen muss zwei Nachfolger haben.



Die eckigen Knoten sind doofer Ballast und werden fast nie dargestellt!

Aufgabe 2)

Wenn wir den baum so durchlaufen würden wie eben, welche Ausgabe würde sich ergeben?

Aufgabe 3)

Es gibt noch zwei weitere Möglichkeiten einen Binärbaum zu durchlaufen. Findet sie heraus und schreibt das Ergebnis auf.

Es gibt wie beim Stack und der Queue festgelegte Operationen auf einem Baum.

Eine haben wir bereits kennen gelernt:

„Suchen“ : Von der Wurzel ausgehend wird überprüft, ob der jeweilige Knoten bzw. sein Schlüssel kleiner oder größer als der gesuchte Knoten ist. Wenn er kleiner ist, wird im linken Teilbaum weitergesucht, wenn er größer ist, im rechten.

Die Suche ist abgeschlossen, wenn der gesuchte Knoten gefunden ist (positiv), oder wenn der nächste zu überprüfende Teilbaum leer ist (negativ).

Aufgabe 4)

Die Operationen „Einfügen“ und „Löschen“ fehlen noch. Überlegt euch, wie man diese Funktionen realisieren könnte und welche Probleme, gerade beim löschen auftreten können.