Einleitende Beispiele

1. Unter sechs gleich aussehenden Kugeln befindet sich eine Kugel mit einem größeren Gewicht. Die anderen Kugeln haben gleiches Gewicht. Mit wie vielen Wiegevorgängen mit einer Balkenwaage mit beliebig großen Waagschalen findet man sicher die schwerere Kugel?

2. Die Anzahl der Kugeln sei n und die Mindestanzahl der Wiegevorgänge sei w.

   (a) Fülle die folgende Tabelle aus.

n	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
w	1															

   (b) Wie viele Wiegevorgänge benötigt man für 1537 Kugeln?

   (c) Freiwillig: Gib eine Formel für den Funktionsterm $w(n)$ der Funktion w an. Tipp: $[x]$ (lies: GAUSSklammer x) ist die größte ganze Zahl $a <= x$.

3. Beschreibe, wie man in einem Lexikon in Buchform nach einem Begriff sucht.

Problemstellung

Ein Array sei schon sortiert. In diesem Feld soll nach einem bestimmten Element gesucht werden. Im schlimmsten Fall muss man dann das ganze Feld von Anfang bis Ende durchlaufen, wenn man den schon behandelten Algorithmus zum Suchen benutzt.
Ein viel effizienteres Verfahren ist die *binäre Suche:*

1. Vergleiche das zu suchende Element mit dem Element in der „Feldmitte“.

2. Sind sie gleich, ist man fertig. Sonst weiß man nun, in welcher „Hälfte“ des Feldes man weitersuchen muss.

3. Wiederhole das Verfahren für die entprechende „Hälfte“ u.s.w.

Anschaulich

Das Feld sei mit a bezeichnet. Den Suchbereich legt man mit den beiden Variablen l (für *low*) und h (für *high*) fest. m steht dann für die „Mitte“ des Feldes. Mit `int m=(l+h)/2` erhält man hier 4 (Integerdivision!). Gesucht werden soll hier nach 13.

i	0	1	2	3	4	5	6	7	8	9
$a[i]$	2	3	5	7	11	**13**	17	19	21	25
	l				m					h

Vergleich mit $a[m]$ zeigt, dass das gesuchte Element im oberen Teil liegen muss. Dementsprechend werden nun die Suchgrenzen verändert und m neu bestimmt.

i	0	1	2	3	4	5	6	7	8	9
$a[i]$	2	3	5	7	11	**13**	17	19	21	25
						l		m		h

i	0	1	2	3	4	5	6	7	8	9
$a[i]$	2	3	5	7	11	**13**	17	19	21	25
						l m	h			

Fertig!

Führe eine Suche nach dem nicht vorhandenen Element 12 durch und trage jeweils l, h und m ein.

i	0	1	2	3	4	5	6	7	8	9
$a[i]$	2	3	5	7	11	13	17	19	21	25
	l				m					h

Programmierung

1. Formuliere mit Hilfe einer geeigneten Schleife eine Methode

   `int binaereSuche(int zahl, int von, int bis)`,

   die die Position von `Zahl` in einem Feld bzw. einem Teil des Feldes mit den Indexgrenzen `von` und `bis` oder beim Fehlen dieser Zahl -1 zurück liefert.

2. Formuliere eine entsprechende rekursive Methode

   `int binaereSuche(int zahl, int von, int bis)`.

Effizienz

Schätze durch eine Formel ab, wie viele Vergleiche bei der linearen und bei der binären Suche im schlimmsten Falle nötig sind, wenn das Feld n Elemente enthält.