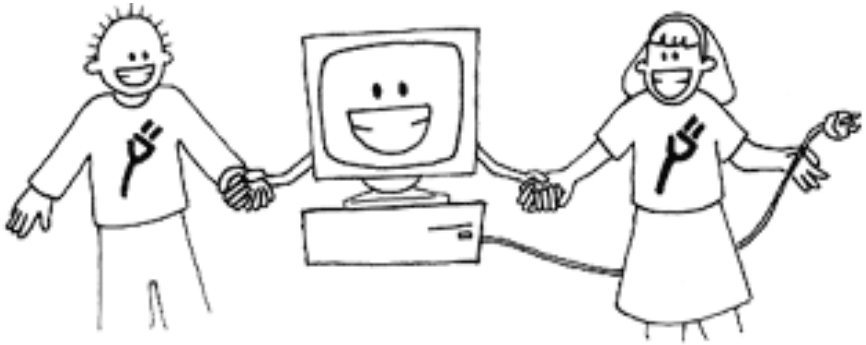
Informatik ohne Rechner

Computer Science Unplugged

Ute Schmid und Bernd Pachur

Fakultät WIAI, Otto-Friedrich Universität Bamberg



29.10.2007

Was können Computer?

- Computer sind heute fast überall, nicht immer sehen sie wie ein normaler Computer aus (Monitor, Tastatur, Gehäuse mit Prozessor, Speicher etc.)
- Computer werden heute von fast allen benutzt, nicht nur von Informatikern oder anderen Fachleuten
- Informatik (auf englisch *computer science*) beschäftigt sich mit den Fragen
  - wie Computer arbeiten
  - wie Computer "denken" (rechnen)
  - wie man Computer besser und schneller machen kann

Aufgabe 1

- Wo werden Computer überall eingesetzt? (Handy, Fernseher, Krankenhaus, Waschmaschine, Supermarkt, ...)

Computer-Algorithmen

- Algorithmus: Anweisungen, die dem Computer sagen, *was* er mit Daten *wie* machen soll
  Aufgabe und *Methode*
- Beispiel: Kochrezept
- Algorithmen sind zentral dafür, dass Computer Aufgaben schnell und korrekt erledigen
- Manche Algorithmen lösen ein Problem besser als andere
- Immer wieder werden Algorithmen entdeckt, die Probleme lösbar machen, die vorher nicht in sinnvoller Zeit auf dem Computer gelöst werden konnten (Berechne PI mit Millionen Nachkommastellen, Finde alle Seiten im WWW, die Deinen Namen enthalten, Finde die beste Möglichkeit, Pakete in einen Container zu packen)

Kochrezepte

200g Butter

200g Zucker

6 Eier

125g Mehl

1 Pck. Backpulver

100 g Kaba

200g Nüsse

1 TL Zimt

Daten

Butter und Zucker schaumig schlagen

Nach und nach Eier dazugeben

Gesiebtes Mehl mit Backpulver, Kaba, Nüssen und Zimt vermengen und unterheben

In eine Springform geben.

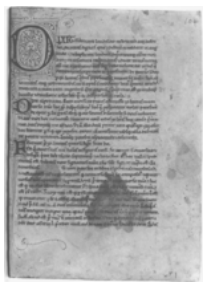
50–60 Minuten bei 170 Grad backen.

Algorithmus

Ergebnis

Wortherkunft Algorithmus

- Das Wort Algorithmus ist eine Abwandlung des Namens von Muhammed Al Chwarizmi (ca. 783 – 850), dessen arabisches Lehrbuch "Über das Rechnen mit indischen Ziffern" (um 825) in der mittelalterlichen lateinischen Übersetzung mit den Worten *Dixit Algorism* begann. Im Mittelalter wurde daraus *algorismus* als Bezeichnung für die Kunst des Rechnens mit den arabischen Ziffern.



Der erste Computer-Algorithmus

- Der erste für einen Computer gedachte Algorithmus wurde 1842 von Ada Lovelace in ihren Notizen zu Charles Babbages Analytical Engine, festgehalten. Sie gilt deshalb als die erste Programmiererin. Weil Charles Babbage seine Analytical Engine nicht vollenden konnte, wurde Ada Lovelaces Algorithmus nie darauf implementiert.

Suchen

- Schnelles Finden von Information ist zentral dafür, dass ein Computer Aufgaben insgesamt schnell bearbeiten kann
- Fast alle Aufgaben beeinhalten Suche
- Beispiel: Finden der Nummer einer Freundin im Addressbuch des Handys
  Beispiel Supermarkt: Finde den Preis eines Produkts (über Strichcode identifiziert) in einer Liste

Aufgabe 2

- Etwa 15 Mädchen halten eine Karte mit einer zufälligen Zahl verdeckt vor sich.
- Ein Mädchen bekommt 5 Süßigkeiten und hat die Aufgabe eine vorgegebene Zahl durch möglichst wenig Fragen zu finden. Jede Frage kostet eine Süßigkeit.
- Wiederholung mit sortierten Zahlen.

Aha Erlebnis 1

- Es gibt verschiedene Methoden (Algorithmen) um zu suchen.
- Sind die Daten vorsortiert geht es im Durchschnitt schneller, wenn man die Fragen "geschickt" einsetzt.

Lineare Suche

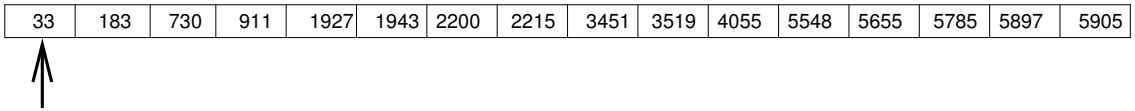
Algorithmus: Lineare Suche

- `Daten: Feld von natürlichen Zahlen`
  1. `Setze den Feldzähler` i `auf 0`
  2. `Solange die gesuchte Zahl nicht gefunden ist:`
     1. `Erhöhe den Feldzähler` i `um 1`
     2. `Vergleiche die aktuelle Zahl mit der gesuchten Zahl`
     3. `Wenn die aktuelle Zahl gleich der gesuchten Zahl ist,`
        `Dann sage ‘‘Zahl auf Position` i `gefunden’’`

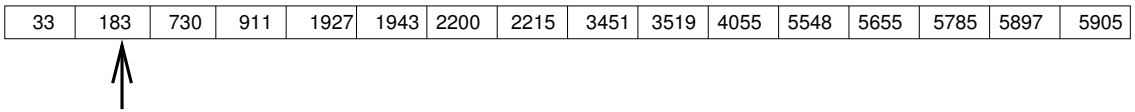
Gesucht: 1927
Feldzähler i=0

33	183	730	911	1927	1943	2200	2215	3451	3519	4055	5548	5655	5785	5897	5905

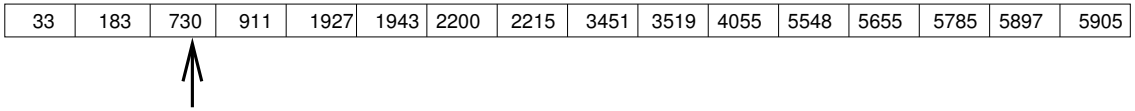
i = 1
33 = 1927?



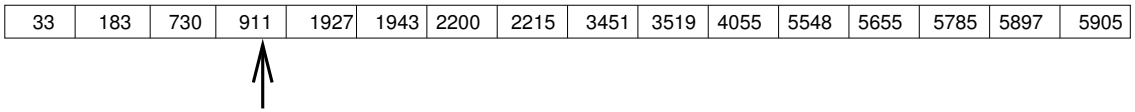
i = 2
183 = 1927?



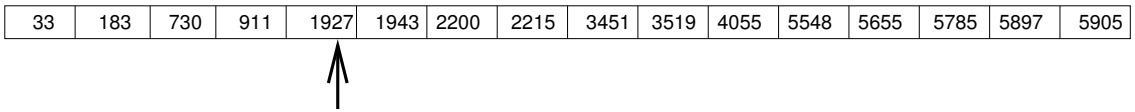
i = 3
730 = 1927?



i = 4
911 = 1927?



i = 5
1943 = 1927?



- Zur Erinnerung: gesucht ist beispielsweise ein Produkt über seinen Strichcode.
  Finde die zu 1927 zugehörige Information (Warenart, Preis, Lagerbestand)
- Anmerkung: Lineare Suche ist für beliebige Zahlenfolgen (ungeordnet) definiert. Wir werden den Algorithmus für sortierte Zahlenfolgen testen, um ihn besser mit einem schlaueren Algorithmus, der nur für sortierte Zahlenfolgen funktioniert, vergleichen zu können.

Illustration

Aufgabe 3

- Paarweise "Schiffe Versenken" mit dem linearen Algorithmus
- Wie lange dauert es im längsten Fall, bis die Zahl gefunden ist, wenn sie vorkommt? (Länge des Feldes n)
  Wieviele Zahlen muss man anschauen, um sicher zu wissen, dass eine Zahl nicht vorkommt?
- Wann kann man sich bei einer sortierten Liste sicher sein, dass die gesuchte Zahl nicht in der Liste ist? (Erweitere den Algorithmus entsprechend)

Aha Erlebnis 2

Der selbe Algorithmus kann auf verschiedene konkrete Probleme (Suche verdeckte Zahl, Schiffe versenken) angewendet werden.

Wie verarbeitet ein Rechner Algorithmen?

- Formulierung in einer Sprache, die der Rechner "versteht" (Computerprogramm)
- Prozessor führt die Anweisungen des Programms auf Daten aus.

Aufgabe 4

- Simulation der Ausführung eines Programms auf dem Rechner
- Ein Mädchen ist Prozessor (setzt den Feldzähler und vergleicht), ein Mädchen gibt die Anweisungen, die anderen Mädchen sind die Daten

Computer vs. Mensch

- Algorithmen werden von Menschen erdacht um Probleme zu lösen.
- Ein Algorithmus kann im Alltag ausgeführt werden (Suchen im Telefonbuch) oder auf dem Computer.
- Damit der Computer den Algorithmus ausführen kann, muss er in einer Sprache geschrieben sein, die er "versteht" – in einer Programmiersprache (das in einer Programmsprache formulierte Programm wird dann automatisch in sogenannte Maschinensprache übersetzt).
- Der Prozessor "liest" den Algorithmus Anweisung für Anweisung und führt die aktuelle Anweisung auf den Daten aus.

Es geht schlauer

Aufgabe 5

- Fällt Euch ein Algorithmus ein, mit dem man schneller die gesuchte Zahl finden kann?
- Hinweis: wie sucht ihr einen Namen im Telefonbuch? ein Stichwort im Lexikon?

Binäre Suche

Algorithmus: Binäre Suche

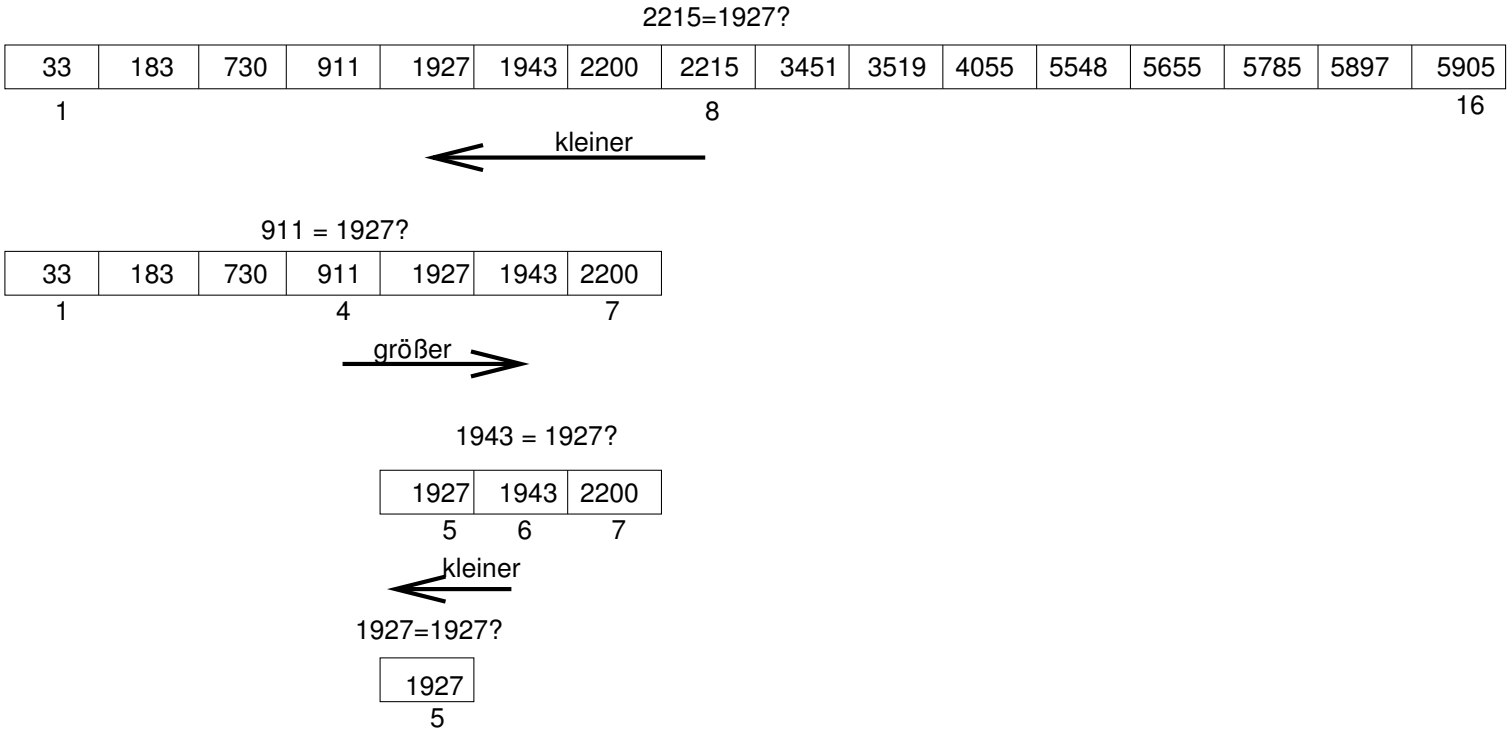
- Daten: Feld von n geordneten natürlichen Zahlen
  1. Sei s die Nummer für den Anfang des Feldes und e die Nummer für das Ende des Feldes.
  2. Sei i der Zähler für das aktuelle Feld.
  3. Setze s am Anfang auf 0 und e auf n.
  4. Solange die gesuchte Zahl nicht gefunden ist:

Binäre Suche

Algorithmus: Binäre Suche

1. (*)Vergleiche die aktuelle Zahl mit der gesuchten Zahl
2. Wenn die aktuelle Zahl gleich der gesuchten Zahl ist,
   Dann sage ‘‘Zahl auf Position i gefunden’’
   Sonst
3. Wenn die gesuchte Zahl kleiner ist als die aktuelle,
   Dann betrachte nur noch das Feld von Position s bis $\frac{e}{2} - 1$ und mache mit Anweisung (*) weiter
   Sonst betrachte nur noch das Feld von Position $\frac{e}{2} + 1$ bis e und mache mit Anweisung (*) weiter.

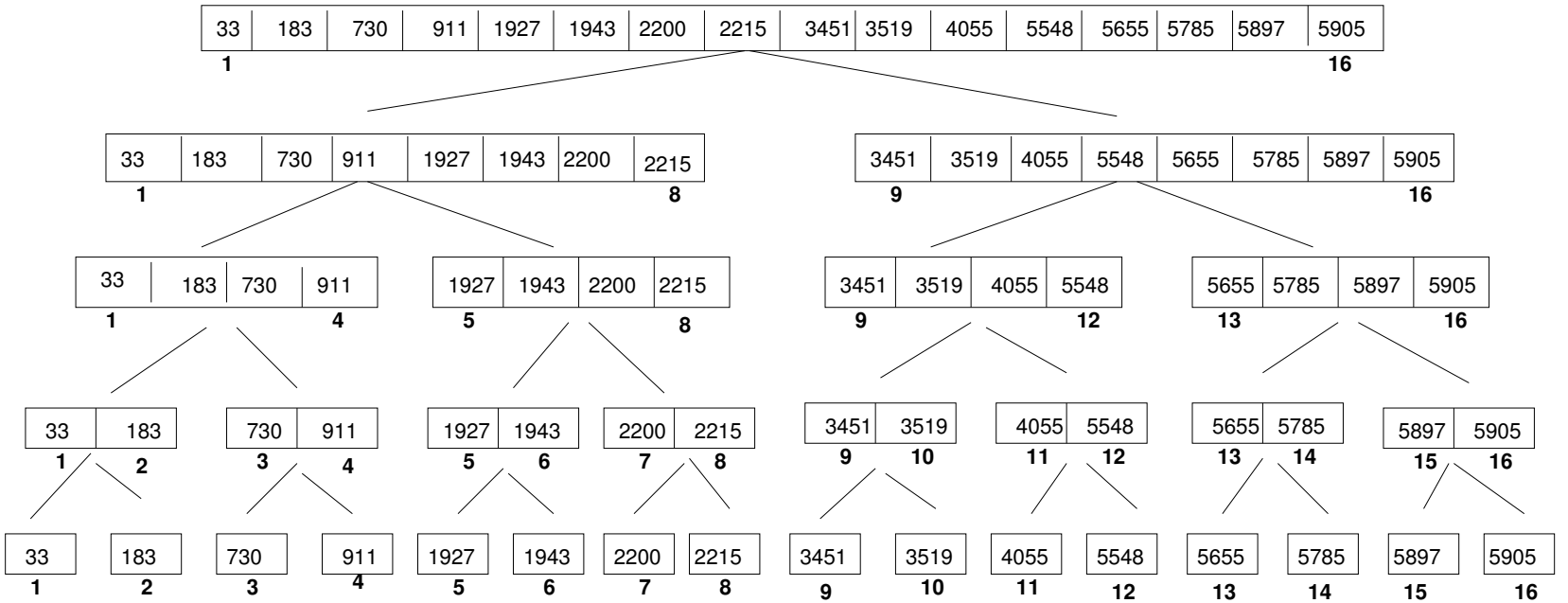
Gesucht: 1927



- Je nach dem, wie man bei ungeraden Längen die neue Mitte berecht, werden folgende Zahlen im jeweils genannten Suchschritt gefunden:
- direkt: 2215
- 1 Schritt: 911, 5548
- 2 Schritte: 183, 1943,3519, 5785
- 3 Schritte: 33, 730, und alle übrigen Zahlen
- danach: Zahl definitiv nicht enthalten

Aufgabe 6

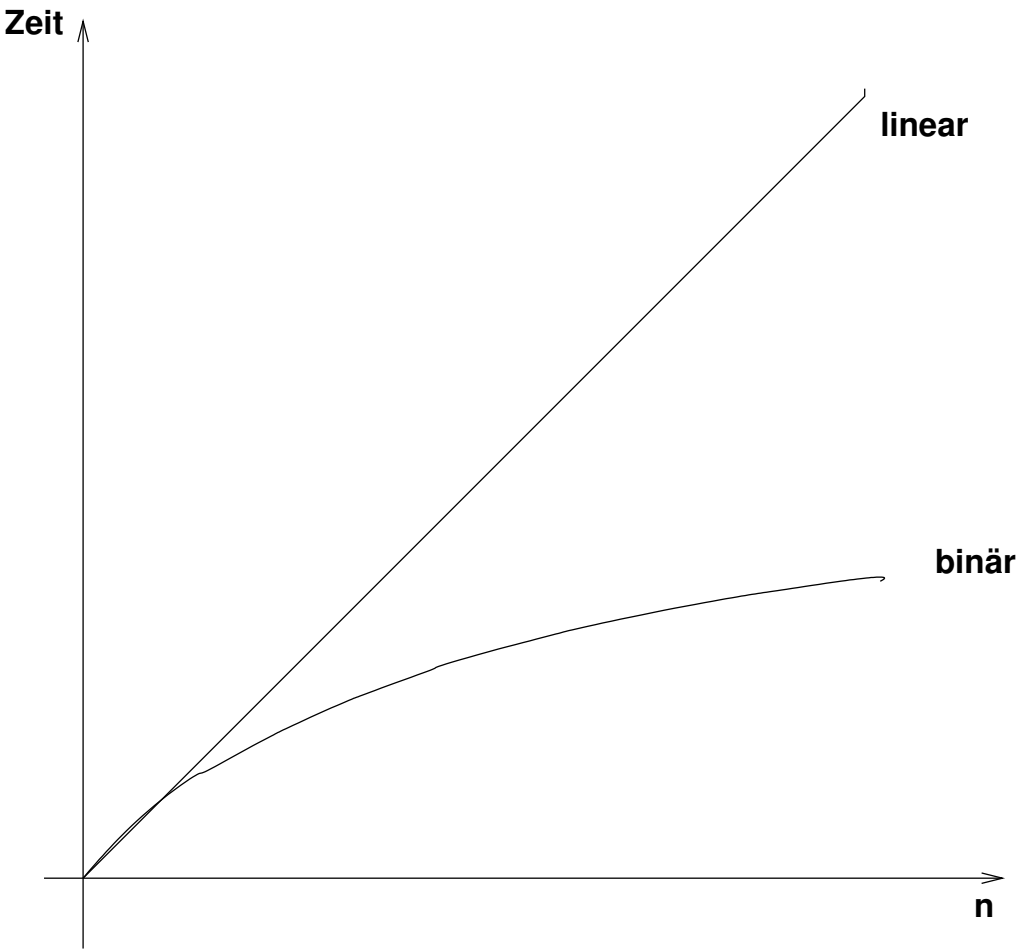
- Spielt paarweise Schiffe versenken und zählt jeweils, wie viele Versuche Ihr braucht, bis die Zahl gefunden ist.
- Hat jemand eine Idee, wie lange man im Durchschnitt braucht? (Herleitung: Entscheidungsbaum)
- Wie schnell wird im Beispiel die Zahl 730 mit linearer Suche gefunden, wie schnell mit binärer?
- Bei wievielen Zahlen im Beispiel ist die lineare Suche schneller, bei wievielen die binäre Suche?
- Nach wieviel Fragen kann man sich bei einem Feld aus 8 Zahlen sicher sein, dass die gesuchte Zahl nicht dabei ist?
- Wendet das Gelernte an, wenn Ihr eine Zahl zwischen 0 und 100 raten sollt.


33 183 730 911 1927 1943 2200 2215 3451 3519 4055 5548 5655 5785 5897 5905
1
16
1 Frage
33 183 730 911 1927 1943 2200 2215
1
8
3451 3519 4055 5548 5655 5785 5897 5905
9
16
2 Fragen
33 183 730 911
1
4
1927 1943 2200 2215
5
8
3451 3519 4055 5548
9
12
5655 5785 5897 5905
13
16
3 Fragen
33 183
1
2
730 911
3
4
1927 1943
5
6
2200 2215
7
8
3451 3519
9
10
4055 5548
11
12
5655 5785
13
14
5897 5905
15
16
4 Fragen
33
1
183
2
730
3
911
4
1927
5
1943
6
2200
7
2215
8
3451
9
3519
10
4055
11
5548
12
5655
13
5785
14
5897
15
5905
16

Aha Erlebnis 3

- Für 100 Zahlen benötigt binäre Suche ungefähr 6 Versuche
- Für 1000 Zahlen etwa 9 Versuche
- Für 1.000.000 Zahlen etwa 19 Versuche
- Die Fragenanzahl wächst viel langsamer als die größe des Feldes!
- Binäre Suche ist effizienter als lineare Suche!
- (Für Fortgeschrittene: $ld(n)$ versus n)

Linear vs Binär



Sortieren

- Eine typische Aufgabe für Computer ist es, große Listen von Dingen zu sortieren: Namen nach Alphabet, EMail nach Datum, Zahlen nach Größe
- Sortieren hilft uns, Information schnell zu finden
- Beispiel: wenn Schulaufgaben nach Noten sortiert sind, sieht man das beste und das schlechteste Ergebnis auf einen Blick
- Wenn man die falsche Methode zum Sortieren verwendet, kann es sehr lange dauern, bis eine große Liste sortiert ist, sogar auf einem sehr schnellen Computer.
- Glücklicherweise existieren sehr viele schnelle Methoden.
- Als Bechränkung für alle Methoden gilt, dass Computer immer nur zwei Zahlen miteinander vergleichen können.

Aufgabe 7

Überlegt alle möglichen Bereiche, in denen es wichtig ist, dass Dinge sortiert sind. Was würde passieren, wenn die Dinge unsortiert wären? (Bücherei, Kontoauszüge, Spielkarten, Telefonbuch ...)

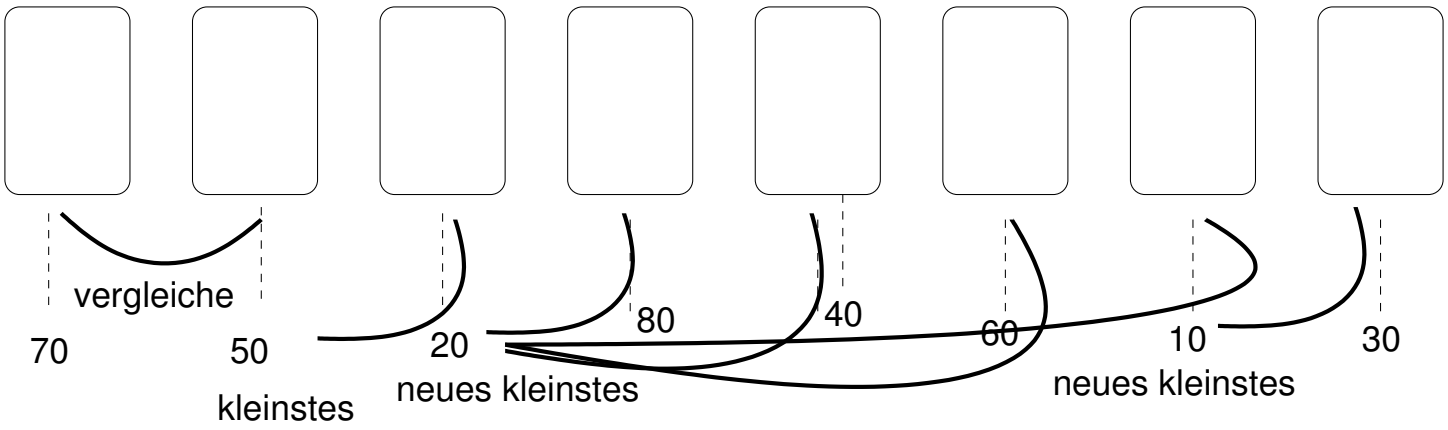
Aufgabe 8

- Ein oder mehrere Mädchen sollen mittels einer Balkenwaage herausfinden, welche von 8 Schachteln die leichteste ist.
- Danach arbeiten die Mädchen in Paaren mit verdeckten Karten, auf deren nicht sichtbarer Seite Zahlen stehen und sollen die kleineste Zahl finden.
- Diskussion der Strategie.
- Nächster Schritt: Sortieren aller Zahlen.

Finden des kleinsten Elements

Aha-Erlebnis 4

- Ohne die Gewichte oder Zahlen zu kennen, kann man das kleinste Element finden, indem man jeweils das bisher leichtere/kleinere von zweien behält.



Sortieren durch Auswählen

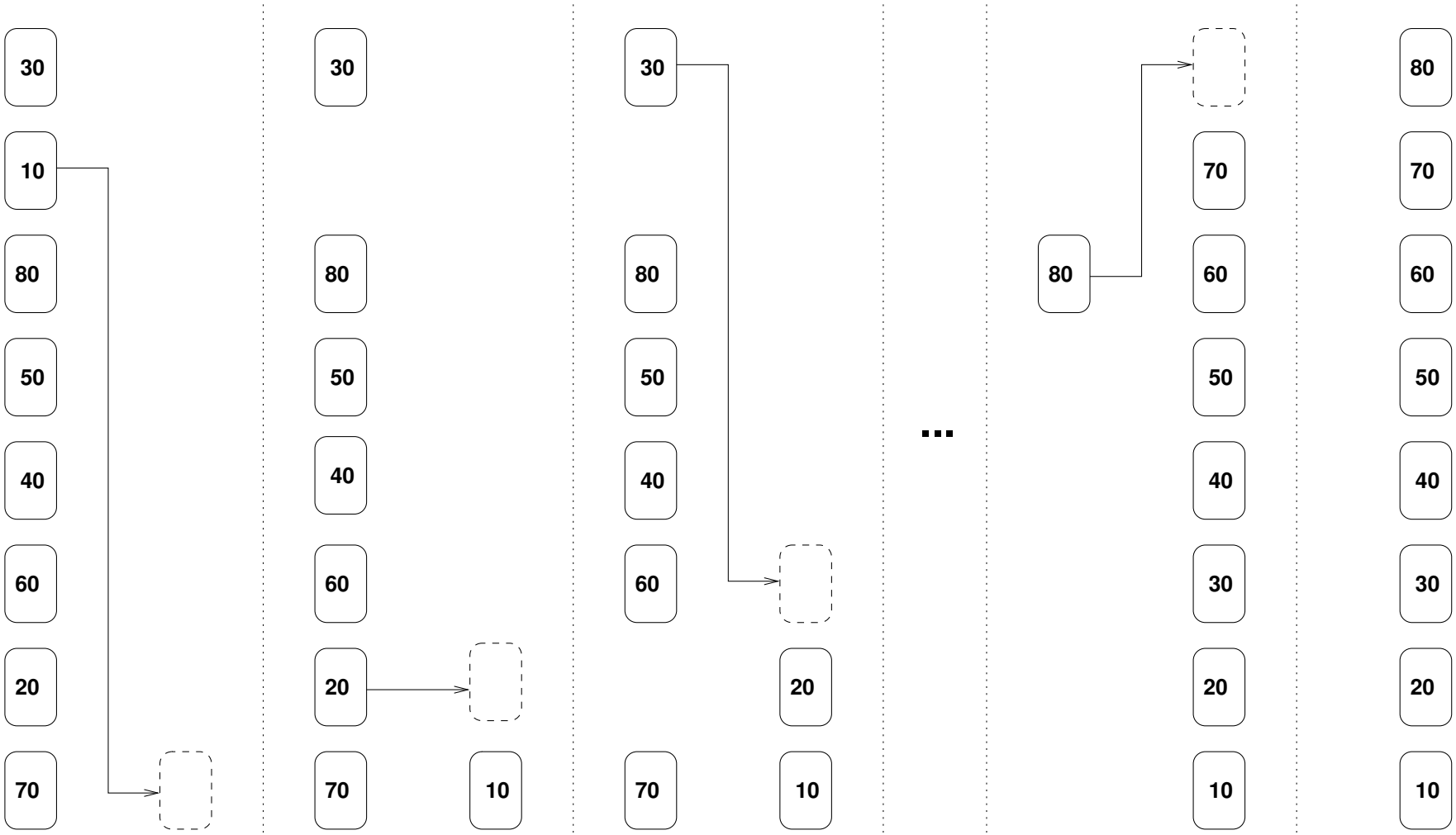
Algorithmus: *selection sort*

1. `Finde die kleinste Zahl und lege sie auf eine Seite`
2. `Mache dasselbe mit den verbliebenen Zahlen, so lange bis keine mehr übrig sind.`

Sortieren durch Auswählen

Finde das kleinste Element durch Vergleich von Paaren:

1. Nimm die Zahl ganz links als bisher kleinste Zahl an.
2. Gehe schrittweise nach rechts durch alle Zahlen.
3. In jedem Schritt: Prüfe, ob die neue Zahl kleiner als die bisher kleinste ist.
4. Wenn ja: nimm die neue Zahl als bisher kleinste.
5. Wenn nein: behalte die alte Zahl.


30
10
80
50
40
60
20
70
30
80
50
40
60
20
70
10
30
80
50
40
60
20
70
10
...
80
70
60
50
40
30
20
10
80
70
60
50
40
30
20
10

Aufgabe 9

- Paare von Mädchen sortieren eine Menge von Zahlen mit *selection sort*
- Zählt wie lange Ihr jeweils braucht, um das kleinste Element zu finden.
- Bei 8 Zahlen: erst 8-mal, dann 7-mal, dann 6-mal, etc. bis nur noch eine Zahl übrig ist.
- Bei zwei Elementen: 1, bei drei: $2 + 1 = 3$, bei vier: $3 + 2 + 1 = 6$, bei fünf: $4 + 3 + 2 + 1 = 10$, bei sechs: $5 + 4 + 3 + 2 + 1 = 15$, bei sieben: $6 + 5 + 4 + 3 + 2 + 1 = 21$, bei acht: ? (28).

Größenvergleiche

Aufgabe 9

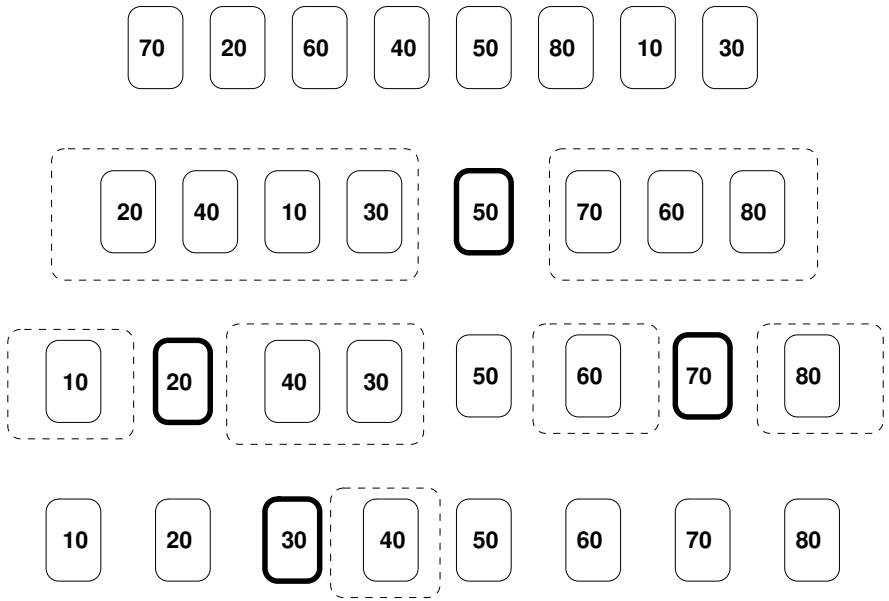
- Bei natürlichen Zahlen reicht es, bei jedem Zahlenpaar (x, y) zu prüfen, ob gilt $x < y$.
- Überlegt Euch, welche Vergleichsoperation man beim Sortieren von Namen und von Datumsangaben benötigt.

Aha-Erlebnis 5

- Es gibt verschiedene Möglichkeiten, Werte zu vergleichen. (Der Fachbegriff heißt Ordnungsrelation.)
- Bei Zahlen: Es reicht der "kleiner" Vergleich
- Bei Datum: Vergleiche erst Jahr, dann Monat, dann Tag
- Bei Namen: Vergleiche erst den ersten Buchstaben des Nachnames, dann den zweiten, etc.; bei gleichen Nachnamen: vergleiche die Vornamen auf die gleiche Art. Welcher Buchstabe "kleiner" als ein anderer ist ergibt sich aus der Reihenfolge im Alphabet.

Quicksort

- Quicksort (englisch: schnelles Sortieren) ist viel schneller als Sortieren durch Auswählen.
- Idee:
  - Wähle zufällig eine der Zahlen und lege sie in die "Mitte".
  - Vergleiche alle anderen Zahlen mit dieser Zahl und lege die kleineren nach links, die größeren nach rechts. (Durch Zufall kann es sein, dass auf einer Seite viel mehr Zahlen landen als auf der anderen.)
  - Wiederhole diese Prozedur für die linke Gruppe und für die rechte Gruppe. (Beachte, dass die jeweils für die Mitte gewählte Zahl nicht mehr mit betrachtet wird.)
  - Wenn in jeder Gruppe nur noch eine Zahl vorhanden ist, ist die Liste sortiert.



Denke dran: Du kannst erst während eines Vergleichs feststellen, ob eine Zahl kleiner als eine andere ist! (Der Computer kann die Zahlen in einer Liste auch nicht alle gleichzeitig “sehen”.)

Aufgabe 10

- Paare von Mädchen sortieren mit der Quicksort-Methode. Zählt, wie viele Schritte Ihr braucht, bis eine Liste sortiert ist.
- Vergleicht die Schrittzahl mit Sortieren durch Auswählen.
- Sortieren durch Auswählen braucht für eine feste Anzahl von Zahlen immer gleich viele Schritte zum Sortieren (z.B. 28 für 8 Zahlen). Überlegt Euch, wieviele Schritte Quicksort für 8 Zahlen braucht.

Aha-Erlebnis 6

- Quicksort kann sehr schnell sein, wenn durch Zufall immer die Zahl für die Mitte ausgewählt wird, die auch die “mittlere” Zahl ist. (Bei 8 Zahlen 14 Vergleiche).
- Wenn man Pech hat und zufällig immer die kleinste Zahl wählt, ist Quicksort genau so langsam wie Sortieren durch Auswählen.

Teile und Herrsche

- Die pfiffige Idee, die Quicksort (im Mittel) so schnell macht, heißt Teile und Herrsche.
- Die Zahlen werden so lange in immer kleinere Gruppen aufgeteilt (teile), bis das Problem ganz leicht zu beherrschen (herrsche) ist.
- Das Teile und Herrsche Prinzip wird durch Rekursion realisiert: Man macht genau dasgleiche immer wieder.
- Sortieren wird damit (im Mittel) sehr schnell

Weitere Sortier-Algorithmen

- Es wurden sehr viele verschiedene Sortieralgorithmen erfunden.
- Einige davon sind:
  - Sortieren durch Einfügen (insertion sort): Nimm ein Element aus der unsortieren Liste und füge es an den richtigen Platz in einer neuen Liste ein (die anfangs leer ist). (Diesen Algorithmus verwenden Kartenspiele häufig, um die Karten in der Hand zu sortieren.)
  - Sortieren durch Vertauschen (bubble sort): Gehe immer wieder durch die Liste und vertausche zwei benachbarte Elemente, wenn sie in der falschen Reihenfolge sind.
  - Sortieren durch Verschmelzen (merge sort): Ist ähnlich schlau wie quicksort und benutzt auch das Teile und Herrsche Prinzip. Die Liste wird immer wieder in zwei Hälften geteilt und die sortierten Teillisten werden dann wieder zusammengepackt.

Schwierige Probleme

- Bislang haben wir gesehen, wie man in Listen von Dingen suchen oder solche Listen sortieren kann.
- Es gibt kompliziertere Gebilde als Listen. In der Informatik spricht man von Graphen.
- Einen Graph kannst Du Dir wie ein Straßennetz vorstellen: Orte sind “Knoten”, Straßen sind “Kanten”.
- Auch in einem solchen Straßennetz kann man suchen. Zum Beispiel könntest Du Dich fragen, in welcher Reihenfolge Du eine Menge von Orten besuchen solltest, damit Du dabei möglicht wenige Kilometer fährst.

Problem des Handlungsreisenden

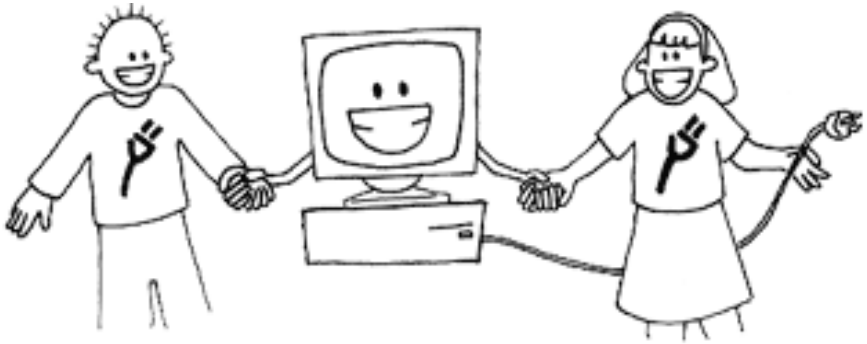
- Das Problem des Handlungsreisenden ist es, eine Rundreise von einem festen Ort aus zu planen. Zum Beispiel: Von Bamberg aus sollen die Orte Augsburg, Ulm, Coburg, Berlin und Frankfurt besucht werden.In Bamberg ist Start und Ziel. Jeder andere Ort soll nur einmal besucht werden. Gesucht ist die Strecke mit den wenigsten Kilometern.
- Ein einfacher Algorithmus ist: Probiere alle Reihenfolgen aus und merke Dir die Kilometerzahl. Wähle die Reihenfolge mit der geringsten Kilometerzahl.
- Das Problem ist, dass dieser Algorithmus bei vielen Orten so lange rechnet, dass wir es nicht erleben würden, bis das Ergebnis feststeht.

Finden möglicht guter Lösungen

- Wenn man einen Rechenschritt mit einer Mikrosekunde veranschlagt, würde man für 25 Städte die beste Strecke in ungefähr einer Minute berechnen können. Für 31 braucht man eine Stunde, für 36 einen Tag, für 39 eine Woche und für 44 ein Jahr!
- Gute Algorithmen für solche schwierigen Probleme zu finden, ist Aufgabe der Forschung in der Informatik.
- Bei solchen schwierigen Problemen muss man häufig darauf verzichten, die beste Lösung finden zu wollen. Stattdessen denkt man sich schnelle Algorithmen aus, die eine möglichst gute Lösung (ziemlich kurze Strecke, aber nicht garantiert die kürzeste) liefern.

Literaturhinweise

- Der Workshop basiert auf Material aus dem Buch:
  Tim Bell, Ian H. Witten und Mike Fellows (2005). Computer Science Unplugged. Zum freien download unter:
  `http://csunplugged.com/`



(leider bislang nicht in Deutsch verfügbar)

Literaturhinweise

- Jens Gallenbacher (2006). Abenteuer Informatik. IT zum Anfassen von Routenplaner bis Online-Banking. Elsevier.