Über Algorithmen

Lernziele

obligatorisch

- du kannst mit einem kollaborativen Dokument arbeiten
- du entwickelst gemeinsam mit anderen eine Lösungsstrategie für ein Sortierproblem
- du setzt gemeinsam mit anderen ein dir bisher unbekannte Lösungsstrategie um
- du kennst den Fachbegriff Algorithmus, seine Merkmale und kannst Bezüge zum Bubblesort-Algorithmus herstellen
- du kannst mit einer abstrakten Syntax den Algorithmus Bubblesort auf Papier umsetzen
- du kennst die Lösungsstrategie „divide & conquer“ als einen grundlegenden Ansatz in der Informatik
- du kennst mit dem algorithmischen Problemlösen einen Teilbereich der Informatik

fakultativ

- du setzt den Quicksort-Algorithmus gemeinsam mit anderen um
- du kannst mit einer abstrakten Syntax den Algorithmus Bubblesort und Quicksort auf Papier umsetzen
- du vergleichst Quicksort und Bubblesort hinsichtlich ihrer Effizienz

Deine Erfahrung

Aufgabe 1 (alleine) - Deine Ideen zur Informatik

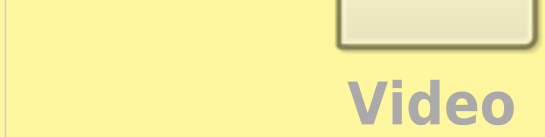
Was ist für dich Informatik? Tippe deine Ideen in die Box unter dieser Aufgabenstellung. Recherchiere dafür **nicht** vorher im Internet. Es geht erstmal um deine Ideen.

Bestimmt ist dir beim Tippen in dem Textfeld schon aufgefallen, dass einiges anders ist. Aber: Herzlichen Glückwunsch! Du hast soeben ein informatisches System (Etherpad) in Form eines kollaborativen Dokuments benutzt.

Mehr zu Etherpad

Tipp 1: Mehr über Etherpad erfahren

Wahrscheinlich kennst du Etherpad schon aus deinem Unterricht - für alle anderen gibt es hier eine kurze Einführung.



Informatik löst Probleme

Aufgabe 2 (zusammen, zwei Gruppen) - Ein einfaches Problem

Stellt euch in eurer Gruppe **möglichst schnell** nach der Schuhgröße auf. Beschreibt in dem unteren Kasten genau, wie ihr dieses Problem gelöst habt. Die andere Gruppe sollte nach eurer Anleitung ohne weitere mündliche Erklärungen eure Strategie später umsetzen können. Das prüfen wir dann auch ...

Ihr habt eine klassisches informatisches Problem bearbeitet - es muss sehr oft etwas sortiert werden, z.B.

- Artikel in Onlineshops nach dem Preis
- Klassenlisten nach Alphabet
- McDonalds-Filialen nach der Entfernung von euch
- [...]

Eine formale Sortierstrategie

Vielleicht habt ihr gemerkt, dass das mit dem Umsetzen von (fremden) Anleitungen gar nicht so einfach ist. Dafür muss man sehr präzise sein.

Aufgabe 3 (zusammen, zwei Gruppen) - Ein Problem nach einer Vorgabe lösen

Ihr bekommt alle eine Zahl auf einem Zettel.

1. Ihr **merkt** euch diese Zahl und lasst sie irgendwo unsichtbar in einer Tasche in eurer Kleidung verschwinden.
2. Ihr stellt euch **irgendwie in einer Reihe** auf
3. Der/die Erste (Person A) in der Reihe sagt seinem linken Nachbarn (Person B) seine Zahl. Person B sagt ihre Zahl ebenfalls.

4. **Wenn** die Zahl von Person A größer ist, als die von Person B, **tauschen** beide die Plätze
5. **Ansonsten** behalten beide ihre Plätze in der Reihe
6. Jetzt wird der/die Zweite in der Reihe Person A und der/die Dritte Person B
7. **Wenn** ihr bei der vorletzten Person der Reihe angekommen seid, fangt ihr wieder bei der ersten Person an
8. Ihr seid fertig, **wenn** ihr einmal durch die gesamte Reihe wandern könnt, ohne dass Plätze getauscht werden
9. Prüft das am Schluss noch einmal, indem ihr in der Reihe stehen bleibt und alle eure Zahlen zeigt

Diskutiert die Unterschiede zwischen dieser Sortierstrategie und eurer eigenen aus Aufgabe 2!

Du hast mit der Beschreibung einen ersten Algorithmus kennengelernt! Diesen gibt es tatsächlich und er nennt sich "Bubblesort".

Lernen: Merkmale eines Algorithmus wissen, erklären und anwenden können

Ein Algorithmus ist eine Vorschrift zur Lösung eines Problems. Er hat folgende Eigenschaften:

1. Das Verfahren muss in einem endlichen Text eindeutig beschreibbar sein (**Finitheit**).
2. Jeder Schritt des Verfahrens muss tatsächlich ausführbar sein (**Ausführbarkeit**).
3. Das Verfahren darf nur endlich viele Schritte benötigen (**Terminierung**).
4. Der Algorithmus muss bei denselben Voraussetzungen das gleiche Ergebnis liefern (**Determiniertheit**).
5. Die nächste anzuwendende Regel im Verfahren ist zu jedem Zeitpunkt eindeutig definiert (**Determinismus**).

Algorithmen werden in der Informatik mit Programmiersprachen umgesetzt. Das sind Sprachen, mit denen sich Lösungsstrategien sehr gut und eindeutig formulieren lassen. Diese Sprachen unterscheiden sich voneinander in ihrer Syntax und ihren Möglichkeiten. Wie du noch sehen wirst, muss so eine Sprache noch nicht einmal Text enthalten - einfache Symbole oder gar Farben erfüllen auch diesen Zweck.

Bubblesort in der Programmiersprache Python

Du kannst dir hier die Umsetzung von Bubblesort in Python ansehen - einer momentan sehr weit verbreiteten Programmiersprache.

bubblesort.py

```
def bubbleSort(arr):
    # Anzahl der Zahlen ermitteln
    n = len(arr)
```

```
    # Erstmal nehmen wir an, dass keine Vertauschungen notwendig waren
    swapped = False
    # Jetzt nehmen wir uns jede Zahl vor und vergleichen sie mit ihrer
Nachbarzahl
    for i in range(n-1):
        # "for" ist eine Schleife, die (n-1)-mal läuft
        # i ist ein Zähler und erhöht sich bei jedem Durchlauf um 1
        # die letzte Zahl lassen wir beim Vergleich aus
        # wir gehen nur bis zur vorletzten (n-1)
        # das sind praktisch die "Durchläufe"
        for j in range(0, n-i-1):

            # bei jedem Durchlauf müssen wir Schritt für Schritt durch
die
            # noch nicht geprüften Zahlen gehen
            # Durchlauf 1 - Schritt 1, Durchlauf 1 - Schritt 2 usw.
            if arr[j] > arr[j + 1]:
                swapped = True
                arr[j], arr[j + 1] = arr[j + 1], arr[j]

        if not swapped:
            # wenn wir nichts tauschen mussten, sind wir fertig
            return


# Zahlen, die zu sortieren sind
arr = [64, 34, 25, 12, 22, 11, 90]

bubbleSort(arr)

print("Ausgabe der sortierten Zahlen:")
for i in range(len(arr)):
    print("% d" % arr[i], end=" ")
```

Aufgabe 4 (zusammen, Partnerarbeit) - Ein Problem nach einer Vorgabe lösen

1. Erklärt, warum die Beschreibung von Bubblesort in Aufgabe 3 ein Algorithmus ist!
2. Warum ist es kein Problem, wenn einige Zahlen mehrfach vorkommen?
3. Wie viele Schritte braucht man bei Bubblesort im einfachsten Fall (bereits sortierte Zahlen) immer?
4. **Schwierig:** Was könnte man ändern, damit der Bubblesort-Algorithmus nie terminiert?

Bubblesort nutzt eine sehr grundlegende informatische Lösungsstrategie, die sich divide&conquer nennt („teile und bewältige“). Es werden immer nur zwei Zahlen (divide) aus einer großen Menge von Zahlen miteinander verglichen (conquer). Das Problem,

welche Zahl größer ist, lässt sich sehr einfach und mit wenig Aufwand lösen.

Bubblesort einmal abstrakt in einer Syntax

Wir nehmen einmal an, dass die Zahlen **3,7,1 und 8** zu sortieren sind.

Durchlauf 1.1: 3 7 1 8 (nichts tauschen)

Durchlauf 1.2: 3 7⇔1 8 (1 gegen 7 tauschen)

Durchlauf 1.3: 3 1 7 8 (nichts tauschen, am Ende angekommen, von vorne)

Durchlauf 2.1: 3⇔1 7 8 (3 gegen 1 tauschen)

Durchlauf 2.2: 1 3 7 8 (nichts tauschen)

Durchlauf 2.3: 1 3 7 8 (nichts tauschen, am Ende angekommen, von vorne)

Durchlauf 3.1: 1 3 7 8 (nichts tauschen)

Durchlauf 3.2: 1 3 7 8 (nichts tauschen)

Durchlauf 3.3: 1 3 7 8 (nichts tauschen, am Ende angekommen und vorher nichts getauscht - wir sind fertig)

Wir brauchen also neun einzelne Vergleiche, um die vier Zahlen zu sortieren. Vergleiche sind die aufwändigste Operation beim Sortieren. Wir müssen in den dritten Durchlauf, da wir noch keinen Durchlauf ohne Tausch hatten. Eine kürzere Schreibweise (**Syntax**) ohne Kommentare könnte so aussehen:

1.1: 3 7 1 8

1.2: 3 7⇔1 8

1.3: 3 1 7 8

2.1: 3⇔1 7 8

2.2: 1 3 7 8

2.3: 1 3 7 8

3.1: 1 3 7 8

3.2: 1 3 7 8

3.3: 1 3 7 8

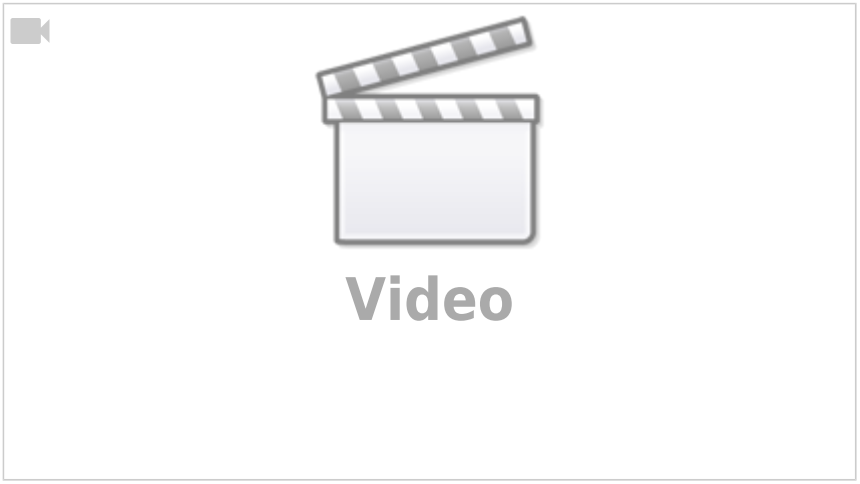
Aufgabe 5 (alleine) - eine Syntax umsetzen

Folgende Zahlenfolge ist gegeben: 3, 7, 1, 8, 2, 5, 9, 4, 6

1. Schreibe die Abfolge der Durchläufe in der kurzen Schreibweise für diese Zahlenreihe auf.
2. Vergleiche zwischendurch immer mal wieder mit den Lösungen deiner Nachbarn.
3. Lösung

Hardcore: Quicksort

Wenn du diesen Abschnitt nicht lösen oder bearbeiten kannst, ist das nicht schlimm. Schau dir dieses Video bis zum Zeitindex 5:00 an.



Es wird dir ein weiterer Sortieralgorithmus mit dem Namen Quicksort angezeigt. Es wird die gleiche Zahlenfolge wie eben sortiert.

- Könntest du das Verfahren in einer Gruppe durchspielen wie bei Bubblesort?
- Was sagt dein Gefühl über die Anzahl der Vergleiche, die für die Sortierung benötigt werden, im Unterschied zu Bubblesort?

Auflösung

Das Aufwändige für einen Rechner sind Vergleiche. Ein Tausch von Zahlen ist vergleichweise einfach - es wird intern nur ein Zeiger umgestellt. Bei Bubblesort hast du gesehen, dass für die gegebene Zahlenreihe 40 Vergleiche notwendig sind, um den Algorithmus abzuschließen.

Quicksort ist etwas schwieriger syntaktisch aufzuschreiben, aber wir schauen mal.

- das aktuelle Pivotelement ist **fett** formatiert.
- eine unterstrichene Zahl ist ein Zahl, die mit dem Pivot-Element verglichen wurde
- die Anzahl der unterstrichenen Zahlen ist also die Gesamtzahl der Vergleiche im aktuellen Durchgang

Durchgang 1 - 6 ist Pivotelement:
3 7 1 8 2 5 9 4 **6**

3 *4* 1 8 2 5 9 7 **6** (4 und 7 wurden getauscht)
3 4 1 8 2 5 9 7 **6**
3 4 1 *5* 2 *8* 9 7 **6**(5 und 8 wurden getauscht)
[3 4 1 5 2] *6* [9 7 8] (6 und 8 wurden getauscht, Vergleich war unnötig, die 6 muss die kleinste Zahl sein)

Wir haben in diesem ersten Durchgang insgesamt 7x Zahlen miteinander verglichen. Die 6 liegt jetzt schon auf ihrer Endposition.

Wir werden weiterhin ...

- für das Pivot-Element **2** 3x vergleichen müssen
- für das Pivot-Element **4** 2x vergleichen müssen
- für das Pivot-Element **8** 1x vergleichen müssen

Wir kommen also mit 13 Vergleichen aus. Quicksort ist um Größenordnungen effizienter als Bubblesort. Bei einer bereits sortierten Liste (Best-Case) bringt Quicksort keinen Nachteil gegenüber Bubblesort.

Ja, und was ist jetzt mit Informatik?

Für eine ganz präzise Formulierung ist es jetzt noch zu früh, aber du hast unglaublich wichtige Grundprinzipien eines *Teilbereichs der Informatik* (algorithmisches Problemlösen) kennen gelernt:

1. **Problemlösungsstrategien** lassen sich in Form eines **Algorithmus** beschreiben
2. Wenn man **Probleme eindeutig und kurz** beschreiben möchte, braucht man dafür eine **Syntax**
3. Es kann **unterschiedliche Lösungsstrategien** für das **gleiche Problem** geben (Bubblesort / Quicksort)
4. Lösungen können **unterschiedlich effizient** sein
5. Man kann **komplexe Probleme** in kleinere, **leichter lösbare Probleme zerlegen** (z.B. Divide&Conquer-Strategie)