Suchen und Sortieren

Didaktische Anmerkungen und Lehrermaterial

Vorbemerkungen

Im niedersächsischen Kerncurriculum Informatik für die gymnasiale Oberstufe[1] wird gefordert, dass die Schülerinnen und Schüler in der Qualifikationsphase unter anderem „das Prinzip, mehrere Daten des gleichen Typs in Reihungen zu verwalten, zu suchen und zu sortieren“ erläutern. Hier wird also bewusst auf die Nennung einzelner Such- und Sortierverfahren verzichtet und stattdessen Wert auf das eigene Entwickeln entsprechender Strategien und Verfahren gelegt. Dies gilt es im Unterricht an vielen Stellen immer wieder umzusetzen: das Fördern prozessbezogener Kompetenzen im Bereich des Algorithmisierens und Implementierens und im Bereich des Kreativen Schaffens und Problemlösens. Es geht nicht darum, spezielle Verfahren auswendig zu lernen, sondern um Kompetenzen wie selbst Algorithmen zu entwickeln, verschiedene Verfahren zu vergleichen oder gegebene analysieren zu können.

Anmerkung für Kurse auf erhöhtem Anforderungsniveau

Sowohl für das Suchen als auch das Sortieren gibt es verschiedene Strategien. Diese unterscheiden sich teilweise sowohl hinsichtlich ihres Zeit- als auch hinsichtlich ihres Speicherbedarfs. Der Vergleich bekannter Such- und Sortierverfahren ist für Kurse auf erhöhtem Anforderungsniveau daher eine gute Übung, um die im niedersächsischen Kerncurriculum aufgeführte Kompetenz „Die Schülerinnen und Schüler beurteilen die Effizienz von Algorithmen unter Abschätzung des Speicherbedarfs und der Zahl der Operationen“ zu festigen.

Suchen

Suchen ist eine Standardaufgabe in der Informatik, die in verschiedensten Kontexten wichtig ist. Im Unterricht bietet es sich an, zunächst Beispiele zu sammeln, bei denen Suchen eine Rolle spielt. Mögliche Schüler:innenantworten hierzu sind etwa Trainingspläne im Fitnessstudio, Ablagestapel auf dem Schreibtisch, Begriffe im Lexikon, Dateien auf dem Rechner und vieles mehr. Für diese Beispiele können die Lernenden Suchalgorithmen beschreiben und diskutieren. Dabei wird klar, dass je nach Daten/Suchkriterium unterschiedlich vorgegangen wird. Einen Zettel im Ablagestapel findet man möglicherweise nur, indem man jedes Dokument einzeln anschaut (also eine lineare Suche durchführt). Dagegen sind Trainingspläne im Fitnessstudio möglicherweise alphabetisch sortiert, so dass man ähnlich zu Begriffen im Lexikon durch geschickte Suchstrategien schneller zum Ziel kommt (also eine binäre Suche durchführt).

In Snap! gibt es bereits Blöcke zum Suchen in Listen (Listen in Snap! entsprechen dynamischen Reihungen): 

bzw. 

, auch andere

[1] vgl. Kerncurriculum für das Gymnasium – gymnasiale Oberstufe, die Gesamtschule – gymnasiale Oberstufe, das Kolleg, Niedersächsisches Kultusministerium, Hannover, 2017, S. 15

Programmiersprachen bieten entsprechende Methoden bereits an. Trotzdem ist die Implementierung eines eigenen Suchalgorithmus eine gute Übung. Die Snap!-Datei `Produktsuche_Vorlage.xml` enthält in der Variablen `Sortiment` eine zweidimensionale Liste von Produkten. Hierfür kann beispielsweise ein Programm zur Suche von Produkten und Ausgabe ihrer zugehörigen Preise implementiert werden. Eine mögliche Lösung finden Sie in der Datei `Produktsuche_Lsg.xml`. Das Materialpaket enthält analog auch eine Vorlage zur Produktsuche und eine mögliche Lösung in Processing.

Sortieren

Genauso wie das Suchen ist auch das Sortieren von Daten eine Standardaufgabe in der Informatik. Häufig werden Daten zunächst sortiert, um eine spätere und schnelle Suche (z.B. eine binäre Suche) darin zu ermöglichen.

Wie eingangs erwähnt sollte beim Sortieren der Schwerpunkt auf den Entwurf und die Implementierung eigener Sortierverfahren gelegt werden. Hierfür bietet es sich an, zunächst haptisch mit geeigneten Materialien (z.B. Zahlenkarten, Quartette, Stifte unterschiedlicher Länge, Haushaltsgegenstände unterschiedlicher Größe) zu arbeiten und damit Sortierverfahren konkret erfahrbar zu machen. Will man zu Beginn bereits die Unterschiede zwischen Mensch und Maschine verdeutlichen, bieten sich auch gleich aussehende Filmdosen mit unterschiedlichen Inhalten, Becher mit eingeklebten Zahlen/Buchstaben oder verschiedene aber gleich aussehende Gewichte an. Auf diese Weise wird ein Gesamtüberblick über alle Werte verhindert und stärker die „Sicht eines Computers“ eingenommen.

Im Folgenden wird ein kurzer Überblick über die infsii-Materialien zum Thema Sortieren gegeben:

Entwicklung eines eigenen Sortierverfahrens

Dokument: Entwicklung_Eigenes_Sortierverfahren

Arbeitsaufträge zur Entwicklung eines eigenen Sortierverfahrens: Zunächst wird in Einzelarbeit ein eigenes Sortierverfahren verbalisiert und erprobt. In einer anschließenden Gruppenarbeitsphase werden verschiedene Strategien verglichen und eine mögliche Formulierung als Algorithmus konkretisiert.

Implementierung eines eigenen Sortierverfahrens

Dokumente:

- Implementierung_Eigenes_Sortierverfahren

Mit diesen Arbeitsblättern wird die Implementierung eines eigenen Sortierverfahrens angeleitet. Zusätzlich werden verschiedene Hilfen zur Unterstützung bei der Implementierung angeboten.

Das Entwickeln eines eigenen Sortierverfahrens ist programmiersprachenunabhängig. Die Implementierung kann in einer beliebigen Programmiersprache erfolgen. Wir stellen im Materialpaket eine Programmvorlage für Snap! und eine für Processing zur Verfügung. Beide Vorlagen enthalten lediglich Methoden zur Erzeugung zufälliger sowie auf- und absteigender Zahlenfolgen, anhand derer das selbst entwickelte Sortierverfahren getestet werden kann. Alternativ können Testdaten auch von den Schüler:innen selbst erzeugt werden. Eine Implementierung in anderen Programmiersprachen ist genauso möglich.

Sie finden Programmvorlagen jeweils für Snap! und Processing in den Dateien:

- Sortieren_Vorlage
- Hilfe: Sortieren_unvollstaendig

Gruppenpuzzle zu unterschiedlichen Sortierverfahren

Dokument: Gruppenpuzzle_Sortierverfahren

Im Anschluss an die Entwicklung und Implementierung eines eigenen Sortierverfahrens können, z.B. durch ein Gruppenpuzzle, bekannte Sortierverfahren untersucht werden. Ziel dabei ist nicht das Auswendiglernen weiterer Algorithmen. Stattdessen ist dies zum einen eine gute Übung zum Nachvollziehen und Erklären gegebener Algorithmen. Zum anderen können im Anschluss diese unterschiedlichen Algorithmen beispielsweise hinsichtlich ihrer Effizienz verglichen werden.

Die methodische Umsetzung als Gruppenpuzzle bietet sich dabei aus mehreren Gründen an:

- Auf diese Weise können in kurzer Zeit viele verschiedene Algorithmen untersucht und verglichen werden.
- Grundsätzlicher Vorteil von Gruppenpuzzles: alle Lernenden sind aktiv beteiligt und müssen einen gegebenen Algorithmus nachvollziehen und ihn anschließend in eigenen Worten erläutern.
- Differenziertes Arbeiten ist sehr gut möglich: Die Lernenden können sich beispielsweise zunächst in einer Stammgruppe treffen und die Algorithmen unter sich aufteilen. Dabei kann vorher der Schwierigkeitsgrad transparent gemacht werden, so dass sich die Lernenden entsprechend ihrer Selbsteinschätzung für ein Sortierverfahren entscheiden können. So sind beispielsweise die iterativen Algorithmen Selection Sort und Bubble Sort als eher leichter verständlich einzuschätzen im Vergleich zu Insertion Sort. Erfahrungsgemäß fällt dagegen das Verständnis der rekursiven Algorithmen Merge Sort und Quick Sort den Lernenden eher etwas schwerer.

Es gibt vielfältige Materialien zu den einzelnen Sortierverfahren, sowohl digital als auch analog. Diese sollten von der Lehrkraft zuvor ausgewählt und den Lernenden zur Verfügung gestellt werden. Die folgende Tabelle gibt ein paar Beispiele. Sie erhebt dabei keinerlei Anspruch auf Vollständigkeit und stellt auch keine inhaltliche Wertung dar[2].

Verfahren	Mögliche digitale Quellen
Selection Sort	• Unter http://abenteuer-informatik.de/dasbuch.html (Link vom 04.03.2021) findet man einen Teil des Buches „Abenteuer Informatik“ von Jens Gallenbacher als Open Access – Version. Darin wird im pdf-Dokument auf S. 69-70 (bzw. laut angezeigter Seitenzahl unten auf S. 57/58) Selection Sort erklärt und zum praktischen „Erleben“ mit beigefügten Bastelmaterialien eingeladen. • http://www.matheprisma.uni-wuppertal.de/Module/Sortieren/index.htm (Link vom 04.03.2021) • https://www.inf-schule.de/grenzen/komplexitaet/sortieren/sortieralgorithmen/selectionsort (Link vom 04.03.2021)

[2] Die Tabelle enthält Links zu externen Webseiten, auf deren Inhalt wir keinen Einfluss haben. Für die Inhalte der verlinkten Seiten ist der jeweilige Webseitenbetreiber verantwortlich.

Bubble Sort	• Unter http://abenteuer-informatik.de/dasbuch.html (Link vom 04.03.2021) findet man einen Teil des Buches „Abenteuer Informatik“ von Jens Gallenbacher als Open Access – Version. Darin wird im pdf-Dokument auf S. 70 unten – 72 oben (bzw. laut angezeigter Seitenzahl unten auf 58 unten – 60 oben) Bubble Sort erklärt und zum praktischen „Erleben“ mit beigefügten Bastelmaterialien eingeladen. • http://www.matheprisma.uni-wuppertal.de/Module/Sortieren/index.htm (Link vom 04.03.2021) • https://www.inf-schule.de/grenzen/komplexitaet/sortieren/sortieralgorithmen/bubblesort (Link vom 04.03.2021)
Insertion Sort	• http://www.matheprisma.uni-wuppertal.de/Module/Sortieren/index.htm (Link vom 04.03.2021) • https://www.inf-schule.de/grenzen/komplexitaet/sortieren/sortieralgorithmen/insertionsort (Link vom 04.03.2021)
Merge Sort	• https://studyflix.de/informatik/mergesort-1324 (Link vom 04.03.2021) • Grafische Veranschaulichung https://de.wikipedia.org/wiki/Mergesort#/media/Datei:Mergesort_example.png (Link vom 04.03.2021), die vollständige Beschreibung des Wikipedia-Artikels ist möglicherweise zu umfangreich für den Unterricht
Quick Sort	• http://www.matheprisma.uni-wuppertal.de/Module/Sortieren/index.htm (Link vom 04.03.2021) • https://www.inf-schule.de/grenzen/komplexitaet/sortieren/sortieralgorithmen/quicksort (Link vom 04.03.2021)