

Didaktische Hinweise

Zeichenkettenverarbeitung mit Processing

Zielgruppe

Der Leitfaden zur Zeichenkettenverarbeitung in Processing¹ richtet sich an Schülerinnen und Schüler in der Einführungsphase, die bereits mit den algorithmischen Kontrollstrukturen und den Grundlagen von Processing vertraut sind. Dieser Leitfaden bietet sich zum Beispiel im Anschluss auf den Leitfaden zum Einstieg in die Programmierung mit Processing an.

Voraussetzungen

Es wird vorausgesetzt, dass die Schülerinnen und Schüler bereits Eingaben des Anwenders mithilfe der Bibliothek *javax.swing.JOptionPane.**; und der Methode *showInputDialog()* erfragt haben. Weiterhin sollten die Schülerinnen und Schüler die Methode *keyPressed()* bzw. *keyTyped()* und die Systemvariable *key* verwenden können, um auf das Ereignis, dass eine Taste auf der Tastatur gedrückt wurde, reagieren zu können. Weiterhin sollten die Schülerinnen und Schüler Ausgaben im Programmfenster, z.B. mithilfe der Methode *text()* erzeugen können. Wenn die Schülerinnen und Schüler zuvor mit dem Leitfaden für den Einstieg in Processing gearbeitet haben, sollten sie über diese Kompetenzen verfügen.

Lernziele

Dieser Leitfaden führt in die Zeichenkettenverarbeitung mithilfe elementarer Zeichenkettenoperationen ein. Die Schülerinnen und Schüler lernen ...

1. ... Benutzereingaben auszuwerten.
2. ... Texte in der Konsole auszugeben.
3. ... auf einzelne Zeichen in einer Zeichenkette zuzugreifen.
4. ... eine Zeichenkette Zeichen für Zeichen zu durchlaufen und zu verarbeiten.
5. ... die ASCII-Codierung der Zeichen bei der Verarbeitung zu verwenden.

Die Schülerinnen und Schüler verwenden dabei die folgenden Zeichenkettenoperationen:

- Bestimmen der Länge einer Zeichenkette
- Auslesen eines Zeichens an einer bestimmten Position
- Verbinden von zwei Zeichenketten zu einer
- Prüfen des Inhalts von zwei Zeichenketten auf Gleichheit

Didaktische Hinweise

Viele der Übungsaufgaben stammen aus dem Bereich der Kryptographie, so dass sich dieser Leitfaden gut mit dem Erlernen der Kompetenzen aus dem Modul Kryptologie kombinieren lässt. Eine Implementierung des Caesar-Verfahrens wird im niedersächsischen Kerncurriculum für die Einführungsphase explizit gefordert. Dies ist daher eine Problemstellung, die sich die Lernenden anhand des Leitfadens Schritt für Schritt erarbeiten können.

¹ Die Programmierumgebung Processing wurde 2001 von Ben Fry und Casey Reas initiiert. Nähere Informationen finden Sie unter <https://processing.org/>

Neben den hier verwendeten Operationen im Umgang mit Zeichenkette sehen die Vorgaben für das Abitur² zwei weitere Operationen vor: Das Ersetzen eines Zeichens an einer übergebenen Position und das lexikographische Vergleichen von Zeichenketten. Da zum Verändern einer Zeichenkette hier die Strategie, diese zeichenweise zu durchlaufen und parallel die veränderte Zeichenkette aufzubauen, zum Einsatz kommt, wird auf eine entsprechende Operation zum Ersetzen von Zeichen verzichtet. Diese kann zu einem späteren Zeitpunkt bei Bedarf eingeführt werden. Da Processing bzw. Java eine solche Methode nicht zur Verfügung stellen, kann diese im Zuge der Einführung von eigenen Methoden selbst erstellt werden (s. Aufgabe 6 im Leitfaden zum Thema „Eigene Methoden“). Bei den vorliegenden Aufgaben wäre sie nur bedingt hilfreich. Das hier verwendete Konzept ist allgemeiner und kann auf eine größere Klasse von Problemen angewendet werden.

Eine lexikographische Ordnung von Zeichenketten erscheint erst sinnvoll, wenn mit mehreren Zeichenketten beispielsweise in einer Reihung gearbeitet wird. Die entsprechenden Java-Methoden *compareTo()* bzw. *compareToIgnoreCase()* können daher im Zusammenhang mit der Einführung von Reihungen eingeführt werden, ggf. erst in der Qualifikationsphase.

Wenn einige Schülerinnen und Schüler eine grafische Sprache bevorzugen, kann das Thema Zeichenkettenverarbeitung parallel in *Snap!*³ bearbeitet werden. *Snap!* stellt die in diesem Leitfaden verwendeten Methoden als Blöcke ebenfalls zur Verfügung. Die angebotenen Materialpakete zur Zeichenkettenverarbeitung in Processing und Snap! sind weitgehend gleich aufgebaut, auch hinsichtlich der Aufgabenstellungen. Weitere Überlegungen zum parallelen Einsatz von Processing und Snap! finden Sie in den didaktischen Hinweisen zur Zeichenkettenverarbeitung mit Snap!

Die Aufgaben

Die Aufgaben üben und festigen die vorhergehenden Erläuterungen. Der Schwierigkeitsgrad variiert. Die Aufgaben 12 und 15d) eignen sich vor allem für besonders leistungsstarke Schülerinnen und Schüler.

Zu Beginn wird ein Überblick über die zur Verfügung stehenden Methoden gegeben. In den Aufgaben 2 bis 4 liegt der Schwerpunkt bei der Auswertung von Nutzereingaben zunächst auf dem Vergleich von Zeichenketten und der Abfrage der Länge einer Zeichenkette. Der Exkurs zum Vergleich von Zeichenketten erläutert, weshalb für den Vergleich von Zeichenketten die Methode *equals()* anstatt des Vergleichsoperators verwendet werden muss. Da eine Unterscheidung zwischen primitiven Datentypen und Objektreferenzen erst für die Qualifikationsphase vorgesehen ist, kann auf die Begründung in der Einführungsphase auch zunächst verzichtet werden.

Mit Beispiel 3 wird dann das systematische Durchlaufen einer Zeichenkette eingeführt. Hier fällt es Lernenden gelegentlich schwer, zwischen dem Wert der Zählvariablen *i* und dem Zeichen an der Position *i* zu unterscheiden. In Zeile 13 des Beispiels 3 wird das *i*-te Zeichen der durchlaufenen Zeichenkette daher in einer separaten Variablen gespeichert, um diesen Schritt explizit zu machen. Zur Veranschaulichung des Ablaufs der *for*-Schleife können die Lernenden zusätzlich zu Abbildung 4

² Niedersächsisches Kultusministerium (Hrsg.) (2021) *Ergänzende Hinweise zum Kerncurriculum Informatik für die gymnasiale Oberstufe am Gymnasium, an der Gesamtschule sowie für das Kolleg*.

<https://cuvo.nibis.de/index.php?p=download&upload=260> [Datum des Zugriffs: 11.04.2025]

³ Snap! wird von der University of California, Berkeley zur Verfügung gestellt: <https://snap.berkeley.edu>

eine Trace-Tabelle erstellen. Das Konzept des zeichenweisen Durchlaufens und Verarbeitens einer Zeichenkette kann mithilfe der Aufgaben 5 bis 9 gefestigt werden.

Schließlich wird die Umwandlung von Zeichen in den zugehörigen Zahlenwert des ASCII-Codes und umgekehrt thematisiert. Diese Umwandlung wird insbesondere zur Implementierung des Caesar-Verfahrens benötigt. Dieses kann daher als Motivation dienen. Da das Caesar-Verfahren aus mehreren Teilschritten besteht und die selbständige Implementierung einigen Schülerinnen und Schülern erfahrungsgemäß schwer fällt, werden in den Aufgaben 10, 11 und 13 zunächst Teilschritte erarbeitet, bevor das Verfahren dann in Aufgabe 15 implementiert wird. Als Hilfestellung kann gemeinsam mit den Lernenden eine Grafik wie in Abbildung 1 entwickelt werden.

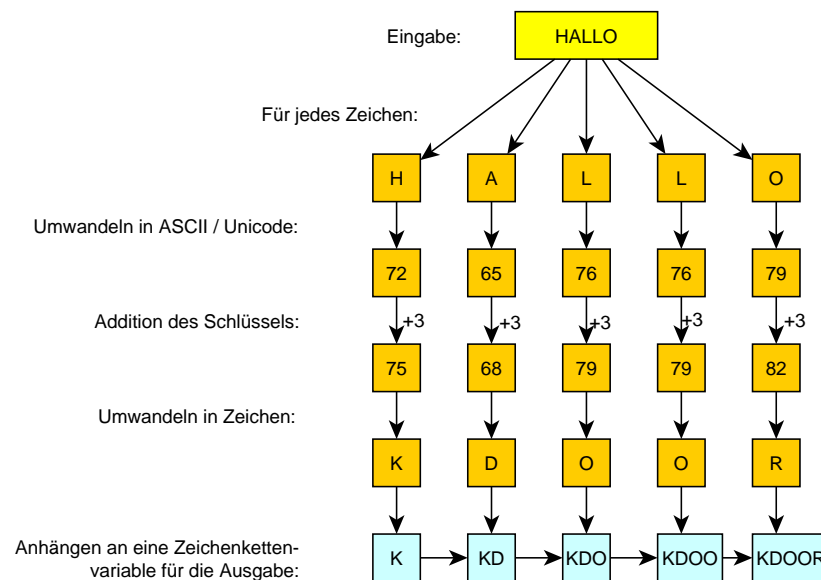


Abbildung 1: Veranschaulichung der Teilschritte beim Caesar-Verfahren

Der Leitfaden kann in einem Projekt münden, in dem die Schülerinnen und Schüler alle erlernten Kompetenzen in Rahmen einer etwas umfangreicheren Problemlösung erproben und anwenden können. Dabei können die prozessbezogenen Kompetenzen aus dem Bereich Kreatives Schaffen und Problemlösen besonders gefördert werden⁴. Das vorgeschlagene Projekt zur Umsetzung verschiedener Verschlüsselungsverfahren eignet sich auch zur Vertiefung und Wiederholung in der Qualifikationsphase. In der Einführungsphase sind hier ggf. einfachere Verfahren zu wählen. Alternativ können Aufgabe 14 und 15 zu einem kleinen Projekt rund um die Caesar-Verschlüsselung inklusive Kryptoanalyse ausgebaut werden.

Lösungsvorschläge können den Ordnern zu der jeweiligen Aufgabe entnommen werden. Dabei ist zu beachten, dass es in der Regel viele verschiedene mögliche Implementierungen gibt und hier nur exemplarisch eine vorgestellt wird.

Ausblick

Im Anschluss an diesen Leitfaden bietet sich z. B. eine Gegenüberstellung der Operationen und Algorithmen zur Zeichenkettenverarbeitung in einer blockbasierten Programmiersprache und in

⁴ vgl. Niedersächsisches Kultusministerium (Hrsg.) (2017) Kerncurriculum für das Gymnasium - gymnasiale Oberstufe, die Gesamtschule – gymnasiale Oberstufe, das Kolleg. Informatik. Hannover: unidruck

Processing an. An diesem Beispiel lässt sich besonders gut verdeutlichen, dass die algorithmischen Konzepte die gleichen sind. Daraus ergibt sich der Vorteil einer programmiersprachenunabhängigen Notation von Algorithmen z. B. in Form von Struktogrammen. Alternativ kann auch mit der Einführung eigener Methoden begonnen werden.

Lizenz

Dieses Werk ist lizenziert unter einer [Creative Commons Namensnennung - Nicht kommerziell - Keine Bearbeitungen 4.0 International Lizenz](#). Sie erlaubt Download und Weiterverteilung des vollständigen Werkes unter Nennung meines Namens, jedoch keinerlei Bearbeitung oder kommerzielle Nutzung.

Für die korrekte Ausführbarkeit der beiliegenden Quelltexte wird keine Garantie übernommen. Auch für Folgeschäden, die sich aus der Anwendung der Quelltexte für die Musterlösungen oder Beispiele des Leitfadens oder durch eventuelle fehlerhafte Angaben ergeben, wird keine Haftung oder juristische Verantwortung übernommen.