

Didaktische Hinweise zur Implementierung einer Huffman-Codierung

Zielgruppe

Die Materialien richten sich vor allem an Lernende in der Qualifikationsphase, die einen Kurs auf erhöhtem Niveau belegt haben, da das Kerncurriculum Informatik für die gymnasiale Oberstufe (s. [1]) in Niedersachsen die Datenstruktur Binärbaum verpflichtend nur als Erweiterung für eA-Kurse vorsieht. Die Lernenden sollten bereits Erfahrungen mit dem algorithmischen Problemlösen unter Verwendung der Datenstruktur Binärbaum und dem händischen Erstellen eines Huffman-Baums haben.

Materialien zur Erarbeitung von Kompressionsverfahren (u. a. der Huffman-Codierung) anhand von Bildern sind in dem Materialpaket „Kompressionsverfahren am Beispiel von Bildern“ enthalten. Materialien zur Einführung und Verwendung der Datenstruktur Binärbaum sind im Materialpaket „Prinzip und Verwendung der Datenstruktur Binärbaum“ zu finden.

Lernziele

Im Modul „Codierung und Übertragung von Daten“ aus dem Lernfeld „Informationen und Daten“ des Niedersächsischen Kerncurriculums für die gymnasiale Oberstufe ist die Anwendung der Huffman-Codierung als Verfahren der Kompression vorgesehen. Da ein Huffman-Code mithilfe eines Binärbaums erstellt wird, bietet sich hier für Kurse auf erhöhtem Niveau im Rahmen eines kleinen Projektes eine algorithmische Umsetzung des Verfahrens an. Dabei wird sowohl der Umgang mit der Huffman-Codierung als auch die Verwendung von Binärbäumen beim Algorithmischen Problemlösen vertieft und gefestigt.

Aufbau der Materialien

Je nach gewählter Reihenfolge der Module kann die Implementierung einer Huffman-Codierung im Rahmen einer Einheit zu Binärbäumen oder zum Thema Kompression erfolgen.

Händisch würde man zunächst einen Huffman-Baum erstellen und diesen dann zur Codierung bzw. Decodierung verwenden. Die Reihenfolge der Aufgaben ist gegenläufig und beginnt mit der Decodierung, da diese im Vergleich zum Codieren oder gar Erstellen eines Huffman-Baums deutlich einfacher zu implementieren ist. Während für die Decodierung ein Binärbaum vom Typ Zeichen ausreichen würde, der in den Blättern die passenden Zeichen speichert, ist es für die Codierung hilfreich, wenn die inneren Knoten eine Zeichenkette speichern, die alle Zeichen der linken und rechten Teilbäume enthält. In den Vorlagen und Lösungen für die Aufgaben 1 und 2 wird daher ein Binärbaum¹ vom Inhaltstyp Zeichenkette verwendet. Eine entsprechende Klasse ist in der Vorlage enthalten. Vorlagen stehen für Processing² und als *JFrame* für den Java-Editor³ zur Verfügung.

Für das Erstellen eines Huffman-Baums ist in Aufgabe 3 zunächst die Entwicklung eines algorithmischen Vorgehens gefordert. Aufgrund der Komplexität der Aufgabe, sollte hier vor der Implementierung ein Ansatz besprochen werden.

¹ Die hier verwendete Datenstruktur Binärbaum orientiert sich an den Vorgaben in [1] und [2]

² Die Programmierumgebung Processing wurde 2001 von Ben Fry und Casey Reas initiiert. Nähere Informationen finden Sie unter <https://processing.org/>

³ Der Java-Editor wird von Gerhard Röhner zur Verfügung gestellt: <https://javaeditor.org/>

Für die Implementierung ist es hilfreich in den Knoten des Binärbaums nicht nur Zeichenketten, sondern Objekte zu speichern, die sowohl eine Zeichenkette als auch eine Ganzzahl enthalten. Für die Verwaltung der Objekte und später der Teilbäume des Huffman-Baums bieten sich dynamische Reihungen an. Die Vorlagen zu Aufgabe 3 enthalten ADT-Klassen mit entsprechenden Inhaltstypen. Im Java-Editor kann alternativ die jar-Datei *adtGesamt.jar*, die von Carsten Rohe für das Landesnetzwerk Informatik implementierte ADT-Klassen enthält, mit entsprechenden Typecasts verwendet werden. Eine Lösung liegt in beiden Varianten vor. Die Vorlagen bereiten eine Umsetzung des algorithmischen Vorgehens vor, dass in der Lösung zu 3a) enthalten ist. Wenn die Lernenden eine andere algorithmische Vorgehensweise entwickeln, wäre die Vorlage anzupassen.

Erweiterungen

Die Implementierung einer Huffman-Codierung kann zu einem größeren Projekt im Bereich Codierung erweitert werden. Zum einen kann eine Huffman-Codierung nicht nur für Texte, sondern auch für Bilder erstellt werden. Wie im Materialpaket „Rekonstruktion ausgewählter Funktionen eines Bildbearbeitungsprogramms“ gezeigt, stellt Processing entsprechende Klassen und Operationen zum Laden und pixelweisen Verarbeiten von Bildern zur Verfügung. Zum anderen können auch andere Kompressionsverfahren (z. B. eine Lauflängencodierung) oder fehlererkennende bzw. fehlerkorrigierende Codes (z. B. Prüfsummen oder (7, 4)-Hamming-Code) implementiert werden.

Literatur

- [1] Niedersächsisches Kultusministerium (Hrsg.) (2017) Kerncurriculum für das Gymnasium - gymnasiale Oberstufe, die Gesamtschule – gymnasiale Oberstufe, das Kolleg. Informatik. Hannover: unidruck
- [2] Niedersächsisches Kultusministerium (Hrsg.) (2021) Ergänzenden Hinweisen zum Kerncurriculum Informatik für die gymnasiale Oberstufe am Gymnasium und an der Gesamtschule sowie für das Kolleg. <https://cuvo.nibis.de/index.php?p=download&upload=260> [Datum des Zugriffs: 09.04.2025]

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 Quelltexte in diesem Arbeitsblatt wird keine Garantie übernommen. Auch für Folgeschäden, die sich aus der Anwendung der Quelltexte oder durch

eventuelle fehlerhafte Angaben ergeben, wird keine Haftung oder juristische Verantwortung übernommen.