Mathematik für die Datenkommunikation
Klasse 3Ib
HS 2020

Fachhochschule Nordwestschweiz
Hochschule für Technik

Programmieraufgabe Huffman

Erstellen Sie ein Programm in Java welches Folgendes leistet:

1. Eine Textdatei (ASCII kodiert) soll eingelesen werden.
2. Es soll eine Tabelle angelegt werden, in der für jedes der 128 möglichen ASCII-Zeichen drinsteht, wie oft das entsprechende Zeichen in der Textdatei vorkommt. (Hinweis: `(int) c` macht aus einem `character c` den entsprechenden ASCII-Wert).
3. Aus dieser Häufigkeitstabelle soll eine Huffman-Kodierung für die Zeichen konstruiert werden, die in der Datei vorkommen.
4. Die Huffman-Kodierungstabelle soll in einer externen Datei `dec_tab.txt` in der Form: ASCII-Code von Zeichen1:Code von Zeichen1-ASCII-Code von Zeichen2:Code von Zeichen2-... gespeichert werden.
5. Die eingelesene Textdatei soll entsprechend der Huffman-Kodierung in einen Bitstring kodiert werden.
6. An diesen Bitstring soll eine 1 und anschliessend so viele Nullen dran gehängt werden, bis der Bitstring eine Länge hat, die ein Vielfaches von 8 ist.
7. Aus diesem erweiterten Bitstring soll ein `byte`-Array erstellt werden, in dem je 8 aufeinanderfolgende Zeichen zu je einem `byte` zusammengefasst werden.
8. Dieses `byte`-Array soll in einer externen Datei `output.dat` gespeichert werden. (Hinweis:

   ```
   FileOutputStream fos = new FileOutputStream("output.dat");
   fos.write(out);
   fos.close();
   ```

   schreibt das `byte`-Array `out` in die Datei `output.dat`,

   ```
   File file = new File("output.dat");
   byte[] bFile = new byte[(int) file.length()];
   FileInputStream fis = new FileInputStream(file);
   fis.read(bFile);
   fis.close();
   ```

   ist der entsprechende Lesevorgang.)
9. Das Programm soll zudem dekodieren können. Es soll also aus externen Dateien die Kodierungstabelle und das `byte`-Array eingelesen werden. Anschliessend soll das `byte`-Array in einen Bitstring umgewandelt werden, von dem dann die letzte 1 und alle folgenden Nullen abgeschnitten werden, um dann den resultierenden Bitstring zu dekodieren und in einer externen Datei `decompress.txt` zu speichern.

10. Testen Sie Ihr Programm an ein paar Textdateien und geben Sie an, wie viel Platz gespart wird.

11. Dekodieren Sie `output-mada.dat` mit `dec_tab-mada.txt`.

Schicken Sie mir bitte sowohl den Quellcode als auch das **Resultat der Dekodierung** von `output-mada.dat` zu.

Allgemeine Hinweise:

1. Sie können in Gruppen von bis zu drei Personen arbeiten.

2. Bei vollständiger Lösung wird auf die Note der MSP 0.2 drauf addiert (falls die Note ≤ 5.8 ist).

3. Sie benötigen keine Kenntnisse über weitergehende Themen der objektorientierter Programmierung. Insbesondere muss keine (kann aber) Klasse für einen Baum geschrieben oder verwendet werden. (Die Huffman-Kodierung könnte mit ein paar Arrays konstruiert werden).

4. Es ist nicht nötig, das Programm hinsichtlich Effizienz zu optimieren.

5. Hilfreich sind diverse Funktionen zur Behandlung von Strings in Java, einen Überblick gibt `http://www.tutorialspoint.com/java/java_strings.htm`.

6. Das Programm sollte verständlich kommentiert sein.

7. Eigentlich gehe ich davon aus, dass Sie aus Fairnessgründen nicht versuchen, zu betrügen. Dennoch werde ich dies (auch mit Hilfe von Tools) kontrollieren. Falls dabei ein Täuschungsversuch festgestellt wird (also: (verschleierte) Kopien von Teilen existierender Programme (Internet oder Kollegen)), wird die Note der MSP auf 1.0 gesetzt.

Abgabe: 18.05.2020