Übungsblatt 6

Aufgabe 1 - Huffman-Codierung

Gegeben sei der folgende Text:
gggbddgffgggcccffeegggcccbcccgggeeffddgggaaaaaddbdddeeffeeffeeffeebeeeffgffbffgffgffgffgggffgggffgggbggg
Bestimme den Huffman-Code für den Text und ergänzen Sie die folgende Tabelle:

Zeichen	Häufigkeit	Huffman-Code
a	5	
b	6	
c	9	
d	10	
e	15	
f	28	
g	32	

In einer 8-bit-ASCII-Codierung würde der Text 840 bits Speicherplatz benötigen.
Wie viele bits benötigt der Text in der oben bestimmten Huffman-Codierung?

Aufgabe 2 - Huffman-Decodierung

Gegeben sei die folgende Codierung: `100100110101111110010001111011101010111000010 1`

Bestimme den zugehörigen Text: _____

Die Codierung der einzelnen Zeichen ist: :

```
token: _ -> code: 101      token: a -> code: 00       token: c -> code: 1000
token: d -> code: 1001     token: h -> code: 11110    token: m -> code: 11111
token: p -> code: 110      token: s -> code: 01       token: t -> code: 1110
```

Aufgabe 3 - Huffman-Dekodierung

Erweitere die Klasse `HuffmanTree` um eine Methode `public String decode( String encoded )`, die den *Huffman*-kodierten Text `encoded` wieder in seine Originaldarstellung bringt. Gehe davon aus, dass die Methode auf genau dem *Huffman*-Baum aufgerufen wird, der auch zur Kodierung des Textes verwendet wurde. Verwende dabei eine Referenz `HuffmanTree current`, die zu Beginn auf die Wurzel des Baums verweist. Während `encoded` zeichenweise gelesen wird, wird die Referenz je nach Zeichen `0` oder `1` auf den linken oder den rechten Kindknoten gesetzt. Wird ein Blatt erreicht, wird der zugehörige Buchstabe dem Ergebnis hinzugefügt und die Referenz wieder auf die Wurzel des Baums gesetzt.

Hinweis: In Java kann mit der Methode `public char charAt( int index )` auf die einzelnen Zeichen eines Strings zugegriffen werden, wobei das erste Zeichen den Index `0` besitzt.