

Universität Leipzig Institut für Informatik Abt. Automatische Sprachverarbeitung	Algorithmen und Datenstrukturen II SS 2009 – Serie 4		
U. Quasthoff, K. Klemm, F. Holz	Ausgabe 04.06.2009	Abgabe 18.06.2009	Seite 1/2

Algorithmen und Datenstrukturen II SS 2009 – Serie 4

1. (11 Punkte) **Kodierung/Huffman**
Gegeben sei folgende Zeichenkette:¹

```
"Lösung "+
" "+
"Satire, störend oft empfunden, "+
"weil stets sie mit Kritik verbunden, "+
"bringt nur bei strikter Unterlassung "+
"Gekränkte nicht mehr aus der Fassung. "
```

Groß- und Kleinschreibung beachten, es gibt keine Zeilenumbrüche, Leerzeichen als Zeichen behandeln, die " und + gehören nicht zur Zeichenkette.

- a) (2 Punkte) Geben Sie zu allen auftretenden Zeichen die Häufigkeiten an. Sortieren Sie die Zeichen dabei absteigend nach ihrer Häufigkeit, Zeichen mit gleicher Häufigkeit werden aufsteigend nach Unicode sortiert.
- b) (6 Punkte) Schreiben Sie ein Programm, welches schrittweise den Code-Baum erstellt. Das Schreiben des Programms ist keine Bedingung. Diese Aufgabe kann auch manuell gelöst werden, wenn auch mit ungleich höherem Aufwand. Geben Sie als Lösung nach dem ersten, zweiten, dritten, vorletzten und letzten Schritt an, welche zwei Knoten/Teilbäume (mit ihren Frequenzen) zusammengefügt wurden (z. B.: Schritt 127: zusammengefügt az 18 und vxy 16 zu neuem Knoten azvxy 34).

Zur Eindeutigkeit der Lösung ist folgende Sortierung der Knotenliste zu benutzen:

- Beginnen Sie mit der Sortierung aus a).
 - Fügen Sie stets die letzten beiden Knoten der sortierten Liste unter Beibehaltung ihrer Reihenfolge zu einem neuen Knoten zusammen.
 - Sortieren Sie den neuen Knoten entsprechend seines Häufigkeitswertes in die Knotenliste ein. Wenn bereits andere Knoten mit dem gleichen Häufigkeitswert existieren, dann wird der neue Knoten entsprechend den Unicode-Werten des ersten Zeichens einsortiert, d. h. ein Knoten cz;= 6 wird vor einem Knoten mbf 6 einsortiert, da $c < m$.
- c) (1 Punkt) Kodieren Sie die ersten 8 Zeichen der obigen Zeichenkette und geben Sie die entstandene Bitfolge an. Fügen Sie zur besseren Lesbarkeit nach jedem kodierten Zeichen ein Leerzeichen zwischen den Bits ein.
- d) (2 Punkte) Geben Sie die Packrate für die Huffman-codierte Zeichenkette an (ohne den Speicherplatz für den Baum), unter der Annahme, daß die ursprünglichen Zeichen UTF-8 kodiert waren.

¹Hansgeorg Stengel, erschienen in H. Stengel: *Ein Dromedar aus Karakum*, Eulenspiegel, Berlin (1999)

Universität Leipzig Institut für Informatik Abt. Automatische Sprachverarbeitung	Algorithmen und Datenstrukturen II SS 2009 – Serie 4		
U. Quasthoff, K. Klemm, F. Holz	Ausgabe 04.06.2009	Abgabe 18.06.2009	Seite 2/2

2. (8 Punkte) **Komprimierung/Lempel-Ziv-Welch**

Gegeben sei folgende Zahlenfolge, die eine LZW-komprimierte Zeichenkette repräsentiert:

5, 8, 18, 2, 7, 4, 17, 18, 29, 5, 17, 31, 33, 38, 40, 18, 8, 35, 19, 35, 43, 8, 19, 25, 4, 50, 18, 18, 19, 55, 46, 32, 34, 55, 42, 39, 51, 46, 17, 48, 63, 7, 30, 61, 65, 10, 0, 3, 4, 11, 11, 4, 13

- a) (6 Punkte) Programmieren Sie den LZW-Algorithmus und entpacken Sie gegebene Wertefolge. Das Alphabet/Lexikon sei dabei auf 128 Zeichen begrenzt. Das initiale Alphabet soll aus den folgend gegebenen Zeichen bestehen:

A	B	C	...	X	Y	Z	Ä	Ö	Ü	□
0	1	2	...	23	24	25	26	27	28	29

Auf ein explizites Endezeichen wird bei dieser Aufgabe verzichtet. Geben Sie als Lösung die entpackte Zeichenkette, sowie den Endzustand des Alphabets/Lexikons an. Geben Sie **nicht** Ihr Programm mit an. Sie können die Aufgabe auch per Hand lösen, allerdings nimmt das mehr Zeit in Anspruch.

- b) (2 Punkte)

Berechnen Sie die Packrate für den LZW-Algorithmus. Dabei seien die Zeichen der ursprünglichen Zeichenkette mit der minimal nötigen Anzahl Bits kodiert. Dasselbe gilt für die Symbole des Lexikons, das der LZW-Algorithmus aufbaut.

3. (7 Punkte) **Stringsuche/Knuth-Morris-Pratt**

Gegeben sei folgende Zeichenkette:

Wilhelm□will□mit□Wilhemine□sprechen

und das Muster:

Wilhemine

- a) (2 Punkte) Geben Sie die Verschiebetabelle (**next-Feld**) für eine Suche des Musters mithilfe des KMP-Algorithmus an.
- b) (4 Punkte) Suchen Sie mithilfe des KMP-Algorithmus in der Zeichenkette nach dem Muster bis zum ersten Match. Geben Sie alle Zwischenschritte als Paare (i, j) , bei denen Mismatches auftreten, an, wobei i die Position in der Zeichenkette und j die im Muster ist. Die Positionszählung beginnt bei 0. Geben Sie außerdem die Position in der Zeichenkette, an der das gefundene Muster beginnt, an.
- c) (1 Punkt) Berechnen Sie die Suchkosten bis zum ersten Match als Anzahl der nötigen Zeichenvergleiche.