- Was ist der Unterschied zwischen den Codierungen ASCII, ISO-Latin-1 und UNICODE?
  → Buchstaben werden als Zeichencodes gespeichert
  → ASCII: Codierung mit 7 Bit, also 128 Zeichen codierbar
  → ISO-Latin-1: Codierung mit 8 Bit, also 256 Zeichen codierbar (ländertypisch)
  → UNICODE: Codierung mit 32 Bit, also 4 Mrd. Zeichen codierbar (Zeichen eindeutig und nicht mehr ländertypisch)

- Was ist UTF-8?
  → Transformationsformat für UNICODE
  → UTF-8 ist das bekannteste und verbreitetste
  → „normale“ Buchstaben entsprechen dem ASCII Code → 0xxx xxxx
  → Sonderzeichen benötigen mehrere Bytes, die Anzahl der Einsen des 1. Byte gibt an wie viele Bytes folgen, Folgebytes beginnen mit 10 → 110x xxxx 10xx xxxx oder 1110 xxxx 10xx xxxx 10xx xxxx

- Sind die folgenden UTF-8 Kodierungen „normale“ Buchstaben oder Sonderzeichen?
  1. 01111001 → y (Buchstabe)
  2. 11000011 10100100 → ä (Sonderzeichen)
  3. 11100010 10000010 10101100 → € (Sonderzeichen)

- Welche verschiedenen Möglichkeiten gibt es um Perl auf UTF-8 Codierung einzustellen?
  → use utf8; → erlaubt Umlaute in Variablennamen
  → use locale; → sortiert nach deutschen Regeln
  → use open “:utf8“; → alle Dateien sind UTF-8 Dateien
  → binmode(STDIN, “:utf8“); → Standardinput wird zur Laufzeit auf UTF-8 gesetzt
  → binmode(STDOUT, “:utf8“); → Standardoutput wird zur Laufzeit auf UTF-8 gesetzt
  → open(INPUT, „<:utf8“, „datei.txt“); → Dateikodierung wird einzeln definiert (Lesen)
  → open(OUTPUT „>:utf8“, „datei.txt“); → Dateikodierung wird einzeln definiert (Schreiben)
  → Starten in der Konsole: perl -C eins.perl → Standardin- und output werden beim Programmstart auf UTF-8 gesetzt

- Was ist die spezielle Variable $_?
  → bekommt automatisch einen Wert zugewiesen
  → z.B. foreach (1..10) { print; }
  → Schleifenwert wird in der Variable $_ gespeichert und von print ohne Argument ausgegeben

- Was ist ein Hash?
  → Hashes gehören zu den Datenstrukturen (Skalare, Listen, neu: Hashes)
  → Hashes sind Paare aus Key (Schlüssel) und Value (Wert)
  → Schreibweise: %hash

- Wie lese ich Key und Value an der Stelle 0 aus?
  → Key: $hash[0] (wie bei Listen)
  → Value: $hash {“Key“}

- Wie erhöhe ich die Anzahl eines $wort im Hash?
  → $hash{$wort}++

- Wie lösche ich ein $wort im Hash?
  → delete (%hash{$wort})

- Wie drucke ich alle Keys/ Values des Hashes aus?
  → print keys %hash
  → print values %hash

- Wie ermittle ich die Anzahl der Hash-Elemente?
  → scalar (keys %hash)

- Wie kann ich testen ob ein Element im Hash existiert?
  → if exists $lexikon{$wort} {…}
  → oder if defined $lexikon{$wort} {…}

- Wie sortiere ich den Hash?
  → sort ANWEISUNG %hash
  → Sortieren nach dem Key: sort keys %hash
  → Alphabetisch sortieren nach dem Wert: sort {$hash {$a} cmp $hash {$b}} keys %hash
  → Umgekehrt alphabetisch nach dem Wert: sort {$hash {$b} cmp $hash {$a}} keys %hash
  → Numerisch sortieren nach dem Wert: sort {$hash {$a} <=> $hash {$b}} keys %hash

- Beispielprogramm zu Hashes (berücksichtigt noch keine Groß- und Kleinschreibung:

```
#!/usr/bin/perl
# Autor: Nicola Greth
# Programm: Beispielprogramm zum Hash
use utf8;
use locale;
use strict;
{
	binmode(STDIN, ":utf8");
	binmode(STDOUT, ":utf8");
	my ($zeile, %lexikon, @woerter, $element);
	print "Bitte geben Sie eine Zeile ein, die in einen Hash
	eingetragen werden soll:>>>\n";
	chomp($zeile = <>);

	@woerter = split(/ /, $zeile);

	foreach $element (@woerter) {
		if (defined ($lexikon{$element})) {
			$lexikon{$element}++;
		}
		else {
			$lexikon{$element} = 1;
		}
	}

	foreach $element (sort {$lexikon{$a}<=>$lexikon{$b}}
	keys %lexikon) {
		print "Wort $element mit der Häufigkeit
		$lexikon{$element}.\n";
	}
}
```