

Programmieren C: Rechnen mit Zeichen, **switch**: Hexzahl-Eingabe

Klaus Kusche

Schreib ein Programm, das mit einer oder mehreren Hexadezimalzahlen auf der Befehlszeile aufgerufen wird und der Reihe nach für jede Zahl den entsprechenden Dezimalwert ausgibt.

Das Programm soll die Umwandlung von einem Wort mit Hexziffern in einen **int** selbst machen (Ziffer für Ziffer!) und keine vordefinierte Eingabe- oder Umwandlungsfunktion dafür verwenden!

Für die Ausgabe des Ergebnisses darfst du ein normales **printf** verwenden.

Die Hexzahlen können entweder 'A' ... 'F' oder 'a' ... 'f' oder beides gemischt enthalten.

Hinweise:

- Du sollst für eine einzelne Hexziffer kein switch mit 16 einzelnen Fällen und Zuweisungen machen, sondern nur 4 Fälle getrennt rechnen:
 - '0' ... '9'
 - 'A' ... 'F'
 - 'a' ... 'f'
 - Ungültiges Zeichen (Fehlermeldung und Programmende!)

Du kannst diese 4 Fälle entweder mit **if**'s oder mit **switch** unterscheiden.
(Zusatzaufgabe: Schreibe zwei Programme, für jede Variante eines!)

Im Fall von **if**: Erwinnere dich: Wie prüft man in einer if-Bedingung, ob eine Zahl oder ein Buchstabe in einem bestimmten Bereich liegt?

In jedem dieser Fälle sollst du aus dem Zeichen den entsprechenden Ziffernwert zwischen 0 und 15 ausrechnen und zum bisherigen Zwischenergebnis dazurechnen (erinnere dich: Wie berechnet man aus dem ASCII-Wert den Ziffernwert?).

Tipps:

- Geh bei jeder Zahl die Ziffern einzeln von links nach rechts durch und rechne ihren Ziffernwert wie oben beschrieben zum Wert der bisherigen Ziffern dazu.

Ein String (Text), zum Beispiel ein Element von **argv**, ist ein Array von einzelnen **char**-Werten. Man kann daher mit [...] auf die einzelnen Zeichen (hier: Ziffern) eines Textes zugreifen, die Nummerierung beginnt wie üblich bei 0 statt bei 1.

Keine Scheu vor zwei [] unmittelbar nacheinander, es funktioniert wie erwartet: Zuerst das x-te Wort von **argv**, und dann das i-te Zeichen aus diesem Wort...

- Wie erkennst du, wie lang ein Text ist, d.h. wie erkennst du, wie viele Ziffern die Hexzahl hat, die deine soeben beschriebene Schleife durchgehen muss?

Es gibt dafür zwei Möglichkeiten:

1. Es gibt eine vordefinierte Funktion **strlen(text)** (aus **string.h**), die als Ergebnis liefert, wie viele Zeichen *text* lang ist.
Die Buchstaben in *text* haben als Index **0** bis **strlen(text) - 1**.
2. Bei jedem Text ist als Ende-Markierung (d.h. nach dem letzten Zeichen) ein **char** mit dem ASCII-Wert **0** (in C als **'\0'** geschrieben) gespeichert.

Du musst also bei jeder Zahl in einer Schleife so lange der Reihe nach von links nach rechts die Zeichen zum Ergebnis dazurechnen, bis das aktuelle Zeichen gleich **'\0'** ist.

Das ist die effizientere Variante, denn **strlen** macht auch nichts Anderes, als mit einer Schleife zeichenweise das **'\0'** zu suchen:
Die Länge eines Strings ist nirgends fertig mitgespeichert.