

Hardwarenahe Programmierung

Musterlösung zu den Übungsaufgaben 7 – 5. Dezember 2024

Aufgabe 1: Hexdumps

Das folgende Programm ([aufgabe-1.c](#)) liest einen String ein und gibt die ASCII-Werte der Buchstaben hexadezimal aus. (Anders als z. B. `scanf()` akzeptiert die Funktion `fgets()` zum Lesen von Strings auch Leerzeichen, und sie vermeidet Pufferüberläufe.)

```
1  #include <stdio.h>
2
3  int main (void)
4  {
5      char buffer[100];
6      fgets (buffer, 100, stdin);
7      for (char *p = buffer; *p; p++)
8          printf ("%02x", *p);
9      printf ("\n");
10 }
```

Beispiel: Bei der Eingabe von `Dies ist ein Test.` erscheint die Ausgabe
`44696573206973742065696e20546573742e0a.`

Schreiben Sie ein Programm, das diese Umwandlung in umgekehrter Richtung vornimmt, also z. B. bei Eingabe von `44696573206973742065696e20546573742e0a` wieder `Dies ist ein Test.` ausgibt.

(6 Punkte)

Hinweis für die Klausur: Abgabe in digitaler Form ist erwünscht, aber nicht zwingend.

Lösung

Siehe [loesung-1.c](#).

Das Programm macht mehrfach davon Gebrauch, daß in C Zeichen und Zahlen äquivalent sind. Wenn z. B. die `char`-Variable `c` den Wert `'3'` (Ziffer 3) enthält, dann hat der Ausdruck `c - '0'` den Wert `3` (Zahlenwert 3). Hierfür ist es insbesondere nicht nötig, vorauszusetzen, daß wir den ASCII-Zeichensatz verwenden und `'0'` den Wert `48` hat.

Bei Eingabe von `44696573206973742065696e20546573742e0a` gibt das Programm zusätzlich eine Leerzeile aus. Die liegt daran, daß das `0a` am Ende bereits eine Zeilenschaltung enthält und das Programm mit `printf ("\n")` eine zusätzliche Zeilenschaltung ausgibt.