

Hardwarenahe Programmierung

Übungsaufgaben – 5. November 2018

Diese Übung enthält Punkteangaben wie in einer Klausur. Um zu „bestehen“, müssen Sie innerhalb von 80 Minuten unter Verwendung ausschließlich zugelassener Hilfsmittel 14 Punkte (von insgesamt 29) erreichen.

Aufgabe 1: Ausgabe von Hexadezimalzahlen

Schreiben Sie eine Funktion `void print_hex (uint32_t x)`, die eine gegebene vorzeichenlose 32-Bit-Ganzzahl `x` als Hexadezimalzahl ausgibt. (Der Datentyp `uint32_t` ist mit `#include <stdint.h>` verfügbar.)

Verwenden Sie dafür *nicht* `printf()` mit der Formatspezifikation `%x` als fertige Lösung, sondern programmieren Sie die nötige Ausgabe selbst. (Für Tests ist `%x` hingegen erlaubt und sicherlich nützlich.)

Die Verwendung von `printf()` mit anderen Formatspezifikationen wie z. B. `%d` oder `%c` oder `%s` ist hingegen zulässig.

(8 Punkte)

(Hinweis für die Klausur: Abgabe auf Datenträger ist erlaubt und erwünscht, aber nicht zwingend.)

Aufgabe 2: Einfügen in Strings

Wir betrachten das folgende Programm (`aufgabe-2.c`):

```
#include <stdio.h>
#include <string.h>

void insert_into_string (char src, char *target, int pos)
{
    int len = strlen (target);
    for (int i = pos; i < len; i++)
        target[i+1] = target[i];
    target[pos] = src;
}

int main (void)
{
    char test[100] = "Hochshule_Bochum";
    insert_into_string ('c', test, 5);
    printf ("%s\n", test);
    return 0;
}
```

Die Ausgabe des Programms lautet: `Hochschhhhhhhhhhh`

- (a) Erklären Sie die Ausgabe. (3 Punkte)
- (b) Schreiben Sie die Funktion `insert_into_string()` so um, daß sie den Buchstben `src` an der Stelle `pos` in den String `target` einfügt.
Die Ausgabe des Programms müßte dann `Hochschule Bochum` lauten. (2 Punkte)
- (c) Was kann passieren, wenn Sie die Zeile `char test[100] = "Hochshule_Bochum";` durch `char test[] = "Hochshule_Bochum";` ersetzen und warum? (2 Punkte)
- (d) Was passiert, wenn Sie `char test[100] = "Hochshule_Bochum";` durch `char *test = "Hochshule_Bochum";` ersetzen und warum? (2 Punkte)

Aufgabe 3: Länge von Strings

Strings werden in der Programmiersprache C durch Zeiger auf **char**-Variable realisiert.

Beispiel: **char** *hello_world = "Hello,_world!\n"

Die Systembibliothek stellt eine Funktion **strlen()** zur Ermittlung der Länge von Strings zur Verfügung (**#include <string.h>**).

- (a) Auf welche Weise ist die Länge eines Strings gekennzeichnet? (1 Punkt)
- (b) Wie lang ist die Beispiel-String-Konstante "Hello,_world!\n", und wieviel Speicherplatz belegt sie? (2 Punkte)
- (c) Schreiben Sie eine eigene Funktion **int strlen (char *s)**, die die Länge eines Strings zurückgibt. (3 Punkte)

Wir betrachten nun die folgenden Funktionen (Datei: [aufgabe-3.c](#)):

```
int fun_1 (char *s)
{
    int x = 0;
    for (int i = 0; i < strlen (s); i++)
        x += s[i];
    return x;
}
```

```
int fun_2 (char *s)
{
    int i = 0, x = 0;
    int len = strlen (s);
    while (i < len)
        x += s[i++];
    return x;
}
```

- (d) Was bewirken die beiden Funktionen? (2 Punkte)
- (e) Schreiben Sie eine eigene Funktion, die dieselbe Aufgabe erledigt wie **fun_2()**, nur effizienter. (4 Punkte)

Viel Erfolg!