

# Hardwarenahe Programmierung

## Übungsaufgaben 4 – 14. November 2024

Diese Übung enthält Punkteangaben wie in einer Klausur. Um zu „bestehen“, müssen Sie innerhalb von 70 Minuten unter Verwendung ausschließlich zugelassener Hilfsmittel 12 Punkte (von insgesamt 24) erreichen.

### Aufgabe 1: Text-Grafik-Bibliothek

Schreiben Sie eine Bibliothek für „Text-Grafik“ mit folgenden Funktionen:

- **void clear (char c)**  
Bildschirm auf Zeichen `c` löschen, also komplett mit diesem Zeichen (z. B.: Leerzeichen) füllen
- **void put\_point (int x, int y, char c)**  
Punkt setzen (z. B. einen Stern `*`) an die Stelle  $(x, y)$  „malen“
- **char get\_point (int x, int y)**  
Punkt lesen
- **void display (void)**  
das Gezeichnete auf dem Bildschirm ausgeben

(8 Punkte)

Hinweise:

- Eine C-Bibliothek besteht aus (mindestens) einer `.h`-Datei und einer `.c`-Datei.
- Verwenden Sie ein Array als „Bildschirm“. Vor dem Aufruf der Funktion `display()` ist nichts zu sehen; alle Grafikoperationen erfolgen auf dem Array.
- Verwenden Sie Präprozessor-Konstante, z. B. `WIDTH` und `HEIGHT`, um Höhe und Breite des „Bildschirms“ festzulegen, z. B.:  

```
#define WIDTH 72
#define HEIGHT 24
```
- Schreiben Sie zusätzlich ein Test-Programm, das alle Funktionen der Bibliothek benutzt, um ein hübsches Bild (z. B. ein stilisiertes Gesicht – „Smiley“) auszugeben.

### Aufgabe 2: Datum-Bibliothek

Schreiben Sie eine Bibliothek (`.c`-Datei und `.h`-Datei) zur Behandlung von Datumsangaben.

Diese soll enthalten:

- einen **struct**-Datentyp `date`, der eine Datumsangabe speichert,
- eine Funktion **void date\_print (date \*d)**, die ein Datum ausgibt,
- eine Funktion **int date\_set (date \*d, int day, int month, int year)**, die ein Datum auf einen gegebenen Tag setzt und zurückgibt, ob es sich um ein gültiges Datum handelt (0 = nein, 1 = ja),
- eine Funktion **void date\_next (date \*d)**, die ein Datum auf den nächsten Tag vorrückt.

Schreiben Sie auch ein Programm, das die o. a. Funktionen testet.

(8 Punkte)

### Aufgabe 3: 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.)