Mikrocontroller, C-Programmierung

Selbststudium und Übungen 2

Martin Vogel, V7.3

Am Ende dieser Semesterwoche können Sie mit Vektoren und Zeigen umgehen.
Sie können einen Vektor dynamisch allokieren und auf dessen Werte zugreifen.

1. Selbststudium

Gehen Sie nochmals die Folien durch.

2. Übungen

(zu diesen Übungen werden durch die Studierenden Musterlösungen präsentiert und abgegeben).

Für die Benutzer-Eingaben bei Übung 2.2 und 2.3 können Sie folgendes Code-Fragment verwenden:

```
    int  wert;

    printf("Wert als Integer (auch negativ): ");
    scanf("%d", &wert);
```

Die Variable `wert` enthält nun den Wert der Eingabe.

Übung 2.1: Reverse

Schreiben Sie eine Funktion `reverse(s)`, die die Zeichenkette `s` umkehrt. Benutzen Sie diese Funktion dazu, ein Programm zu schreiben, das seine Eingabe zeilenweise umkehrt.

Zum Einlesen einer einzelnen Zeile können Sie folgende Funktion verwenden (die Funktion liefert nur 0 zurück, wenn ein 'Ctrl-D' Zeichen oder ein EOF gelesen wird):

```
/**
 * Liest eine Zeile von maximal limit Zeichen ein.
 * Die Zeichen werden (inklusive Zeilenende-Zeichen) im übergebenen
 * Vektor s /0 terminiert abgelegt.
 * @param s     Zeiger auf den Vektor zum Speichern der Eingabe
 * @param limit Maximale Grösse des Vektors
 * @return Anzahl eingelesene Zeichen
 */
int readLine(char s[], int limit) {
    int i = 0;
    int c;

    c = getchar();              /* Buchstabe einlesen          */
    while((c != EOF) &&         /* Ende File ...               */
          (c != '\n') &&        /* oder Ende Zeile ...         */
          (i < limit - 1)) {    /* oder Limite des Speichers?  */
        s[i] = c;
        i++;
```

```
        c = getchar();
    }

    if (c == '\n') {
        s[i] = '\n';     /* Zeilenumbruch anfügen!              */
        i++;
    }
    s[i] = '\0';         /* Zeichenkette-Ende anfügen!          */

    return i;            /* Anzahl gelesene Zeichen zurückgeben */
}
```

Standard Ein-/Ausgabe um- und weiterleiten

- Standard Eingabe aus Datei (anstelle Tastatur)
  `Program  < Dateiname.extension`
- Standard Ausgabe in Datei (anstelle Konsole)
  `Program > Dateiname.extension`
- Ausgabe an ein nächstes Programm als Eingabe weiterleiten
  `Program1 ¦ Program2`

Beispiele:

```
reverse < Reverse.c

reverse < Reverse.c > c.esreveR
reverse < c.esreveR

reverse < Reverse.c | reverse
```

Übung 2.2: Integer to String

Schreiben Sie eine Funktion `char* itoa(int i)`, welche die übergebene Integerzahl in dezimaler Darstellung als Zeichenkette zurückgibt. Die dazu benötigte Zeichenkette soll dynamisch alloziert werden.

Zur Berechnung der Anzahl Zeichen eines Wertes in dezimaler Darstellung können Sie den 10er Logarithmus mit der Funktion `log10(…)` berechnen. Den Absolutwert einer Zahl erhalten Sie mit der Funktion `abs(…)`.

```
#include <math.h>

double log10(double z);
int abs(int z);
```

Hinweis: Vielleicht können Sie die Funktion reverse(…) aus Übung 2.1 gebrauchen.

Übung 2.3: Entwurf Übung mit Vektoren und `malloc`

Entwerfen Sie eine eigene Übung zum Thema Vektoren und `malloc`, analog der weiteren freiwilligen Aufgabe 'Fibonacci'.

Sie entwerfen die Aufgabenstellung und erarbeiten die Musterlösung so wie in Kapitel 3, Weitere freiwillige Aufgaben.

Übung 2.4: Enum to String

Schreiben Sie eine Funktion, die zu dem übergebenen enum-Wert (z.B. color_t) die entsprechende String-Repräsentation zurückgibt.

Deklarieren Sie dazu einen eigenen enum-Typ und verwenden Sie diesen Typ in der Parameterliste der Funktion.

Hinweis: Es gibt kein Sprachelement, das diese Umwandlung bereitstellt.

3. Weitere freiwillige Aufgaben

strcat

Schreiben Sie die Funktion `mystrcat(s, t)` (Sting concatenate) mit Zeigern, welche eine Zeichenkette `t` an das Ende der Zeichenkette `s` kopiert.
Zur Erinnerung: Das `'/0'` Zeichen markiert das Ende einer Zeichenkette.

Für die Eingabe einer Zeichenkette können Sie folgendes Code-Fragment benutzen.

```
char  input[128];

printf("Zeichenkette: ");
scanf("%s", input);
```

Fibonacci Zahlen

Die Fibonacci-Folge f_0, f_1, f_2, …. ist durch das rekursive Bildungsgesetz
$f_n = f_{n-1} + f_{n-2}$ für n >= 2
mit den Anfangswerten
$f_0 = 0$ und $f_1 = 1$
definiert.
Das bedeutet in Worten:

- Für die beiden ersten Zahlen werden die Werte null und eins vorgegeben.
- Jede weitere Zahl ist die Summe ihrer beiden Vorgänger.

Daraus ergibt sich:

f_0	f_1	f_2	f_3	f_4	f_5	f_6	f…
0	1	1	2	3	5	8	…

Schreiben Sie ein Programm, welches den Benutzer abfragt, wie viele Fibonacci Zahlen berechnet werden sollen.
Anschliessend allozieren Sie einen genügend grossen Vektor, berechnen und speichern gemäss obiger Definition alle Fibonacci zahlen und geben Sie sie anschliessend aus.

Musterlösung strcat (Version HCS08):

```
/* Copyright 2015 Hochschule Luzern - Technik & Architektur */

#include <hidef.h> /* for EnableInterrupts macro */
#include "platform.h" /* include peripheral declarations */
#include "clock.h"
#include "sci.h"
#include <stdio.h>
#include <stdlib.h>
#include <termio.h>
#include <libdefs.h>
#include <float.h>

/**
 * Kopiert String t ans Ende von String s.
 * Vorsicht: String s muss genügend gross sein!
 */
void mystrcat(char *s, const char *t) {
    while (*s){             //solange kein \0 gefunden
        s++;
    }
    while (*s++ = *t++) {
    }  ;
}

main() {
    char ch;
    char* str = &ch;
    char  input1[64];
    char  input2[64];

    initClock();
    sci2Init(9600);
    EnableInterrupts;

    (void)printf("Hello Terminal!\n");
    (void)printf("I am the MC car - Press ENTER to start\n");
    (void)printf("=====================================\n");

    (void)scanf("%s",str);

    for(;;) {
        (void)printf("Erste Zeichenkette: ");
        (void)scanf("%s", input1);
        (void)printf("\nZweite Zeichenkette: ");
        (void)scanf("%s", input2);

        mystrcat(input1, input2);

        (void)printf("\nErgebnis von strcat: %s\n", input1);
    }

    (void)printf("program stopped");

    for(;;) {
    }
}
```

Musterlösung Fibonacci (Version HCS08):

```
/**
/*
 * Copyright 2015 Hochschule Luzern - Technik & Architektur
 */

#include <hidef.h> /* for EnableInterrupts macro */
#include "platform.h" /* include peripheral declarations */
#include "clock.h"
#include "sci.h"
#include <stdio.h>
#include <stdlib.h>
#include <termio.h>
#include <libdefs.h>
#include <float.h>

main() {
    char ch;
    int number, i, l;
    int* fibonaccis;

    initClock();
    sci2Init(9600);
    EnableInterrupts;

    do {
        // Wert einlesen
        (void)printf("Wieviele Fibonacci-Zahlen wollen Sie berechnen? ");
        (void)scanf("%d", &number);
        (void)printf("\n");

        l = number + 1;
        fibonaccis = (int*) malloc(sizeof (int) * l);
        if (fibonaccis) {
            fibonaccis[0] = 0;
            fibonaccis[1] = 1;

            for (i = 2; i < l; i++) {
                fibonaccis[i] = fibonaccis[i-1] + fibonaccis[i-2];
            }

            (void)printf("Die Fibonacci Zahlen von 0 bis %d lauten:\n", number);
            for (i = 0; i < l; i++) {
                (void)printf("%d\n", fibonaccis[i]);
            }

            free(fibonaccis);
        } else {
             (void)printf("Fehler in Speicherallokation\n");
        }

    } while(TRUE);
}
```