Einführung in die Programmierung
Aufgabenblatt 4

1) Schreiben Sie ein Programm **`folge.c`**, das eine Folge von reellen Zahlen vom Typ **`float`** einliest. Das Ende der Zahlenfolge wird erkannt durch das erste Zeichen, welches keine Zahl (also z.B. ein Buchstabe) ist. Das Programm soll dann für die eingelesenen Zahlen folgende Größen berechnen:

    1. Die Anzahl,
    2. die Summe,
    3. das Maximum,
    4. das Minimum,
    5. den Mittelwert,
    6. die Standardabweichung (Stichprobenvarianz)

    der eingelesenen Zahlen.

    Die eingelesenen Zahlen seien mit $x_i \ \ (i = 1, \dots, n)$ bezeichnet. Der Mittelwert $\bar{x}$ ist dann definiert durch (1) und die Standardabweichung σ durch (2).

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \qquad (1)$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n-1}} \qquad (2)$$

    Sie werden feststellen, dass bei der Berechnung der Standardabweichung mit der Formel (2) das Problem auftritt, dass man sich alle eingelesenen Zahlen x_i "merken" (d.h. abspeichern) muss. Eine einfache Lösung wäre, eine Feld-Variable zu verwenden. Dies soll hier jedoch vermieden werden. Man kann das Problem ebenfalls lösen, indem man die Standardabweichung mittels

$$\sigma = \sqrt{\frac{\left(\sum_{i=1}^{n} {x_i}^2\right) - n \cdot \bar{x}^2}{n-1}} \qquad (3)$$

    berechnet. Im Gegensatz zu Feld-Variablen lässt diese Formel Zahlenfolgen beliebiger Länge zu. Für die Wurzelfunktion werden noch ein paar Extras benötigt: Zum einen muss der Standard-Header `math.h` eingebunden werden. Die Deklaration der Wurzelfunktion selbst lautet:

```
double sqrt(double);
```

    Zum anderen muss in der Make-Datei beim Linken der Parameter `-lm` angegeben werden, damit die mathematische Bibliothek eingebunden wird:

```
gcc –o ueb04 ueb04.o -lm
```

2) Was ist an folgendem Programm falsch?

```
#include <stdio.h>

int main()
{
    char wort[]={'H','a','l','l','o'};

    printf("%s\n", wort);
    return 0;
}
```

3) Warum ist dieses Programm nicht ganz korrekt?

```
#include <stdio.h>

int main()
{
    int zahlen[10], i = 0;

    for (i = 0; i <= 10; i++)
        zahlen[i] = i;

    return 0;
}
```

4) Was gibt folgendes Programm aus?

```
#include <stdio.h>

int main()
{
    char string[] = "Hans hackt Holz hinterm Hirtenhaus";

    string[10] = '\0';
    string[5] = 'k';

    printf("%s", string);
    printf("%s\n", &string[15]);

    return 0;
}
```

5) Schreiben Sie ein Programm zur Primzahlberechnung nach dem Sieb-Verfahren. Legen Sie dazu alle Zahlen (z.B. von 2 bis 100) in einem Array ab. Beginnend mit der kleinsten Zahl wird diese Zahl als Primzahl auf dem Bildschirm ausgegeben und gleichzeitig werden alle Vielfachen dieser Zahl im Array auf 0 gesetzt (d.h. aus der Liste gestrichen).
Anschließend wird die nächste Zahl ≠0 im Array bearbeitet, usw.

6) Da es in der Programmiersprache C keine dedizierte Ein-/Ausgabe Formatierung für Felder gibt, müssen diese elementweise ein- bzw. ausgelesen werden. Eine Ausnahme gibt es für Zeichenketten mit `"%s"`:

```
char str[10];
scanf("%9s", str);
```

Der Adressoperator & ist bei der Eingabe von Zeichenketten nicht notwendig. Der Wert 9 stellt hierbei sicher, dass nur 9 Zeichen im String abgelegt werden (plus `'\0'`) und damit nicht über das Ende des Feldes im Speicher hinaus geschrieben wird. Es werden also maximal 9 Zeichen von der Konsole gelesen. Verwenden Sie diese Funktionalität für die folgenden Teilaufgaben.

Schreiben Sie eine Funktion **`void`** `invert()`, die als Parameter eine Zeichenkette erhält und alle darin enthaltenen Großbuchstaben in Kleinbuchstaben umwandelt und umgekehrt. Beispiel: Ist s als String vereinbart und initialisiert, so liefert der folgende Code-Abschnitt

```
char s[] = "Hallo Welt";

printf("%s\n", s);
invert(s);
printf("%s\n", s);
```

die Ausgabe:

```
Hallo Welt
hALLO wELT
```

Hinweis 1: Informieren Sie sich zuvor über die Übergabe von Arrays an Funktionen: http://openbook.rheinwerk-verlag.de/c_von_a_bis_z/011_c_arrays_005.htm .
Verwenden Sie zudem die ASCII-Code Tabelle als Hilfe zur Konvertierung von Kleinbuchstaben in Großbuchstaben und umgekehrt.

Hinweis 2: Verwenden Sie statt `scanf()` die Funktion `fgets()`, um einen String mit Leerzeichen einzulesen. Siehe: https://en.cppreference.com/w/c/io/fgets .

7) Schreiben Sie nach dem gleichen Prinzip eine Funktion **`void`** `reverse()`, die die Reihenfolge der in `s[]` enthaltenen Zeichen umkehrt:

```
Hallo Welt
tleW ollaH
```