**Pflichtaufgaben:**

### Aufgabe 1: Datentypen

a) Schreiben Sie ein Programm **wb**, das die Wortbreite der folgenden elementaren Datentypen ausgibt.

- char
- int
- float
- double

Soweit anwendbar, sollen auch die Wortbreiten der mit **short** und **long** modifizierten Datentypen ausgegeben werden.

b) Ergänzen Sie Ihr Programm so, dass die Zahlen int a = 125 und int b = -39 als

- Zeichen
- vorzeichenlose Ganzzahl
- Hexadezimalzahl
- vorzeichenbehaftete Ganzzahl
- Gleitpunktzahl mit einfacher Genauigkeit und 2 Nachkommastellen
- Gleitpunktzahl mit doppelter Genauigkeit und in exponentieller Darstellung

ausgeben werden.

Verwenden Sie für die Ausgabe die Funktion ***printf()***. Eine Funktionsbeschreibung (Linux Manual Page) ist auf dem LEA Server als PDF-Datei abrufbar.

### Aufgabe 2: *Struct Rational*

Viele Operationen mit rationalen Zahlen erfordern neben der Verwendung des ggT (vergleiche Aufgabe 3 Übungsblatt 1) auch die Bestimmung des kleinsten gemeinsamen Vielfachen (kgV).

a) Schreiben Sie eine Funktion, die das kgV von zwei ganzen Zahlen bestimmt.

b) Erstellen Sie ein ***struct Bruch***, der eine rationale Zahl mit Zähler und Nenner repräsentiert. Deklarieren Sie diese Struktur anschließend mittels ***typedef*** als neuen Datentyp "***Rational***".

c) Definieren Sie zu dem neuen Datentyp folgende Funktionen:

```
//Kürzt einen Bruch
Rational kuerze(Rational r)
//Berechnet die Summe von a und b
Rational addiere(Rational a, Rational b)
//Berechnet die Differenz von a und b
```

```
Rational subtrahiere(Rational a, Rational b)
//Berechnet das Produkt von a und b
Rational multipliziere(Rational a, Rational b)
//Berechnet den Quotient von a und b
Rational dividiere(Rational a, Rational b)
//Berechnet aus Zähler und Nenner eine Fließkommazahl
float toFloat(Rational r)
//Gibt einen Bruch auf der Kommandozeile aus
void ausgabe(Rational r)
```

**Aufgabe 3:** Eingabe mit `scanf()`

Schreiben Sie ein Programm, das die in Aufgabe 1 entwickelten Funktionen nacheinander testet. Dazu muss Ihr Programm die Werte für Zähler und Nenner der rationalen Zahlen zur Laufzeit von der Tastatur einlesen. Verwenden Sie dazu die Funktion ***scanf()*** aus ***stdio.h***.

**Empfohlene Aufgaben:**

**Aufgabe 4:**

Erstellen Sie ein Programm **op**, das mit einer per Kommandozeile übergebenen positive float-Zahl x folgende Operationen ausführt und das Ergebnis ausgibt:

| **Operation** | **Ausgabe** |
|---|---|
| Quadrat | float |
| x modulo 3 | int |
| Wurzel | double |
| $\frac{5x+3}{7*(x-1{,}5)}$ | double |
| Bitverschiebung um zwei Stellen nach links | hexadezimal |
| Bitweise UND-Verknüpfung mit der Zahl 0x00F0 | hexadezimal |
| Inkrementieren | int |

**Aufgabe 5:** *Struct Messung*

In einer Werkhalle sollen Messwerte von Produkttests gespeichert werden. Im Folgenden sind die Messwerte und deren Wertebereiche festgelegt:

| | |
|---|---|
| Spannung: | -250 V bis +250 V |
| Stromstaerke: | -10.0 A bis +10.0 A |
| Druck: | 0 Bar bis 20 Bar |

a) Erstellen Sie einen ***struct Messung***, der die oben dargestellten Messwerte speichern kann. Überlegen Sie sich welche Datentypen für die Messwerte geeignet wären.

b) Erweitern Sie die Struktur ***struct Messung*** um einen Zeitstempel. Dazu müssen Sie zunächst die Struktur um eine Komponente vom Typ ***struct tm*** erweitern. Wählen Sie den Namen ‚zeit' für diese Komponente. Der ***struct tm*** ist Teil der C-Bibliothek und kann nach dem Einbinden des Header ***time.h*** (‚#include <time.h>') verwendet werden.

c) Testen Sie ***struct Messung***, indem Sie in ***main()*** eine Variable dieses Typs erstellen und diese mit Werten befüllen. Lesen Sie die Werte für Spannung, Strom und Druck mit der Funktion ***scanf()*** von der Tastatur ein. Die Funktion ***time()*** liefert die aktuelle Zeit in einer Variablen vom Typ ***time_t***. Mit der Funktion ***localtime()*** kann diese in den Typ ***struct tm*** konvertiert werden. Die Funktionen und Typen sind Bestandteil der C-Bibliothek und lassen sich nach dem Einbinden des Header time.h (‚#include <time.h>') verwenden.

```
Beispiel:
time_t t;
time(&t);
struct tm zeit = *localtime(&t);
```

d) Schreiben Sie eine Ausgabefunktion ausgabe(), die den Inhalt einer Variablen vom Typ **struct Messung** in einer lesbaren Form auf der Kommandozeile ausgibt.