C-Kurs E1: Basiswissen

Zielsetzung:
- Umgang mit ganzzahligen Typen und Fliesskommazahlen
- Ausgabe auf die Konsole mit **`printf()`**, Lesen von der Konsole mit **`scanf()`**
- Einführung in Funktionen

Hinweis:
Lösen Sie zuerst die mit (***) bezeichneten Aufgaben: Diese sind die Wichtigsten.

Aufgabenblock A: Ganz- und Fliesskommazahlen, Konsoleneingabe mittels printf()
In diesen Aufgaben wird der Umgang mit ganzzahligen Datentypen, Fliesskommazahlen und deren Ausgabe mit dem **`printf()`**-Befehl beübt.

A 1: (***) **`printf()`**-Befehl für ganze Zahlen
Geben Sie die Zahl **`1000`** in Dezimal-, Oktal- und Hexadezimaldarstellung auf der Konsole aus. Als Formatelement beim **`printf()`**-Befehl werden Sie hierzu als Typ **`d`**, **`o`** und **`x`** einsetzen. Da die restlichen Felder des Formatelements nicht gesetzt werden, heisst das Formatelement z.B. **`%d`**.

A 2: (***) Formatierte Ausgabe von Fliesskommazahlen mit **`printf()`**
Berechnen Sie den Umfang eines Kreises und geben Sie das Resultat auf zwei Kommastellen genau auf der Konsole aus. Der Radius des Kreises soll als globale Variable festgelegt werden.

A 3: (***) Darstellungsarten von ganzzahligen Typen
Schreiben Sie ein Programm, welches die Hexadezimal-Zahl **`ff`** und die Oktal-Zahl **`23`** in Dezimalschreibweise auf der Konsole ausgibt.

A 4: Wertebereich von ganzzahligen Typen
Können die beiden Zahlen **`128`** und **`-128`** mit einem **`signed char`** dargestellt werden?

A 5: Wertebereich von ganzzahligen Typen
Wenn ein vorzeichenbehafteter, ganzzahliger Datentyp die Länge **`n`** Byte hat, ist die grösste positive darstellbare Zahl 2^{n-1}-1, die grösste negative Zahl 2^{n-1}. Warum ist der Menge der positiven Zahlen um eins kleiner als die Menge der negativen Zahlen?

A 6: (***) Speicherbedarf von ganzzahligen Typen
Bei C ist der Speicherbedarf, welcher ein ganzahliger Typ braucht, nicht fix festgelegt, sondern richtet sich nach der Hardware des Rechners. Schreiben Sie ein Programm, welches die Anzahl Byte und Bit für die Datentypen **`char`**, **`short`**, **`int`**, **`long`** und **`long long`** auf der Konsole ausgibt. Setzen Sie hierzu die Funktion **`sizeof()`** ein.

A 7: Fallstrick mit Fliesskommazahlen

Schauen Sie das untenstehende Beispiel an. Eigentlich müsste der Variablen **a** der Wert 0.666... zugewiesen werden. Die Praxis zeigt aber, dass der Variablen **a** als Wert **0** zugewiesen wird. Worin liegt das Problem?

```
double a;
a = 2 / 3;
...
```

A 8: Fallstrick mit Fliesskommazahlen

Gegeben sei das untenstehende Programm. In einer Schleife wird die Variable **a** zehnmal um **0.1** erhöht und anschliessend mit **1.0** verglichen. Wenn man das Programm ausführt, wird die Konsolenausgabe aber nicht ausgelöst. Analysieren Sie das Problem und korrigieren Sie den Fehler.

```
double a=0.0;
int i=0;
for (i=0;  i<10; i++) {
    a = a + 0.1;
}
if ( a == 1.0 ) {
    printf("Das Resultat ist 1.0\n");
}
```

A 9: Wertebereich einer Fliesskommazahl, Aufgabe mit **printf()**

Die Kapazität eines Kondensators wird in Farad [F] angegeben. Im Empfangsteil eines Radios werde ein Kondensator mit der Kapazität 12pF ($12pF = 12*10^{-12}F$) eingesetzt. Kann dieser Wert einer **float**-Variablen zugewiesen werden? Falls ja, schreiben Sie ein Programm, wo Sie diesen Wert in Exponentialdarstellung ausgeben.

Aufgabenblock B: Funktionen

In diesem Block werden einige Aufgaben zum Thema Funktionen gestellt.

A 10: Deklaration einer Funktion

Sie sollen eine Funktion entwickeln, welche die Fakultät n! (5! = 1*2*3*4*5) einer Zahl bestimmt. Wie sieht die Deklaration der Funktion aus?

A 11: (***) Implementation einer einfachen Funktion

Das untenstehende Programm berechnet die Mehrwertsteuer. Schreiben Sie das Programm so um, dass zur Berechnung eine Funktion **calcMwst** eingesetzt wird. Der Mehrwertsteuersatz

mwstsatz soll als globale Variable definiert werden.

```
int main() {
    double mwstsatz = 0.075;
    double wert, mwst;
    wert = 40.00;
    mwst = wert * mwstsatz;
    printf("Die Mehrwertsteuer von %f ist %f\n", wert, mwst);
    return 0;
}
```

A 12: Das untenstehende Programm lässt sich nicht übersetzen, was ist der Grund?

```
int main() {
    dummy();
    return 0;
}
void dummy() {}
```

Aufgabenblock C: Konsolenausgabe mittels scanf()

Bei diesen Aufgaben beschäftigen wir uns damit, Benutzereingaben von der Konsole einzulesen. Hierzu stellt C den Befehl **scanf()** zur Verfügung.

A 13: (***) Benutzereingabe mit **scanf()**

Schreiben Sie ein Programm, welches einen Fliesskommawert von der Konsole entgegennimmt und diesen wiederum mit **printf()** auf der Konsole ausgibt. Bemerkung: dem Befehl **scanf()** müssen Sie eine Referenz übergeben, z.B. **&x**, das heisst, der Funktion wird die Adresse übergeben, wo der Variablenwert gespeichert werden soll.

A 14: (***) Mehrere Variablen mit **scanf()** einlesen

Schreiben Sie ein Programm, welches mit einem einzigen **scanf()**-Befehl drei ganze Zahlen vom Anwender entgegennimmt und diese wieder auf der Konsole ausgibt.

A 15: Stolperstein bei **scanf()**

Der Programmierer hat ein kleines Programm geschrieben, welches zum Ziel hatte, eine Echo-Funktionalität zu programmieren. Wenn der Anwender einen Satz eingibt, soll dieser Satz wieder auf der Konsole ausgegeben werden. Das Programm lässt sich übersetzen, geht aber nicht richtig. Welche Eigenschaft des **scanf()**-Befehls macht da einen Strich durch die Rechnung?

```
int main() {
    char text[80];
    scanf("%s", text);
    printf("%s", text);
    return 0;
}
```