Dr. Michael J. Winckler
IWR, Raum 302
INF 368
69120 Heidelberg
Michael.Winckler@iwr.uni-heidelberg.de
http://www.iwr.uni-heidelberg.de/teaching/Mathe-Star/

Arbeitsblatt 01 *Der C-Compiler: Eine Einführung* (Version 1.10)

Theorie

Übungsziele: Das erste Übungsblatt ist dem Umgang mit dem Compiler und der prozeduralen Programmiersprache C gewidmet.

Im *Syntaxteil* werden Variablennamen und Datentypen erforscht. Die unterschiedlichen Anwendungsformen der einzelnen Datentypen und ihre z.T. begrenzte Darstellungeigenschaften werden an einfachen Beispielen untersucht.

Im *Programmaufbau* betrachten wir monolithische Programme, die auf einer einzelnen Source–Datei beruhen. In dieser Source wird zudem nur eine einzelne Prozedur, daß Hauptprogramm `main()` deklariert. Das Abbild dieser Modularisierung wird in den ersten einfachen Projektdateien sichtbar. Zudem werden wir für eine dazu passende Modularisierung auf Festplattenebene sorgen.

Bei der *Befehlsanalyse* beschäftigen wir uns mit einfachen Befehlen zur formatierten Ausgabe. Neben der C–Funktion `printf()` spielen die zur Ausgabe verwendeten ,,Formatstrings,, und die durch Escape-Sequenzen ansprechbaren Sonderfunktionen der Formatierung eine wichtige Rolle.

Ein weiteres Thema sind die einfachen Mathematikfunktionen und Ihre Wirkungsweise auf Variablen unterschiedlicher Datentypen.

Übung 1.1

Das einfachste Projekt, dem man sich in der Programmierung stets zuerst widmet, ist die Ausgabe einer kurzen Textnachricht.

Aufgabe: Bringe den Text `Hello World!` auf den Bildschirm.

Dieses erste Programm entwerfen wir zusammen am Programmsystem *Borland C-Builder 6*. Danach compilieren wir es, um aus dem für Menschen lesbaren *Sourcecode* den für das CPU verständliche *Executable* zu erzeugen. Dieses erste fertige Programm können wir von einer DOS-Eingabebox (auch Command-Shell genannt) ausführen.

Übung 1.2

Durch die formatierte Ausgabe mit dem Befehl `printf()` kann man die Darstellung von Text (später auch von programminternen Variablen) genau steuern.

Schreibe ein Programm `gedicht.c`, das ein 3–strophiges Gedicht auf den Bildschirm ausgibt. Der Titel des Gedichts sollte etwa in der Mitte des Bildschirms unterstrichen erscheinen. Die drei Strophen sind unterschiedlich weit eingerückt.

```
          Seelische Gesundheit
          --------------------

   Ein Mensch frisst viel in sich hinein:
   Missachtung, Aerger, Liebespein.
```

```
Und jeder fragt mit stillem Graus:
Was kommt da wohl einmal heraus?

    Doch sieh! Nur Guete und Erbauung.
    Der Mensch hat praechtige Verdauung.

                        Eugen Roth
```

Um die versetzte Eingabe zu erzielen, geben wir vor den Textzeilen das Tabulatorzeichen `\t` aus. Zum markieren eines Zeilenendes reicht es nicht aus, den jeweiligen Ausgabebefehl zu beenden! Wir benötigen auch hier ein spezielles Zeichen, das Zeilenendezeichen `\n`, das wir ebenfalls wie einen einfachen Buchstaben ausgeben.

Erweitere dein Programm so, dass du herausfinden kannst, wie lange ein Tabulator in dieser Programmumgebung ist. Mit einer cleveren Programmierung kann man dies *ohne Experimentieren* klären!

Übung 1.3

Schreibe drei Programme, die je drei Variablen der Datentypen `int`, `short int` und `float` deklarieren. Suche dir passende Variablennamen und diskutiere mit deinem Programmierpartner über die Namenswahl.

Initialisiere in deinem ersten Programm zwei der `float`–Variablen mit den Zahlen 3.0 und 7.0. Lasse danach dein Programm das Ergebnis der Addition, Subtraktion, Multiplikation und Division dieser beiden Variablen ausrechnen und ausgeben. Dazu verwenden wir die dritte Variable als Ergebnisvariable und erstellen einen geeigneten Ausgabebefehl. Die Ausgabe könnte in etwa folgendermassen aussehen:

```
Summation von zwei float-Variablen:
    3.0 + 7.0 = 10.0

Subtraktion von zwei float-Variablen:
    3.0 - 7.0 = -4.0
...
```

Wiederhole (in einem neuen Programm!) diese Übung mit Ganzzahl–Variablen. Nimm auch noch den Modulo–Operator % mit hinzu. Was berechnen Divisions- und Modulo–Operator?

Verwende in einem dritten Programm `short int`. Belegen eine deiner Variablen mit 3.0, die andere mit 2.0 und multiplizieren Sie die erste immer wieder mit der zweiten (Code?). Es ergeben sich nacheinander die Ergebnisse 3.0, 6.0, 12.0, ... Gebe diese Ergebnisse in einer Tabelle untereinander aus (neue Zeile: `\n`) und beobachte! Was stellst du fest?

Anm.: Für diese Übung benötigen wir zwei neue Befehle: Die Ausgabe von Variablen auf den Bildschirm lösen wir über *Platzhalter*. Zur mehrfachen Ausführung eines Befehls gibt es *Schleifenkonstruktionen* wie z.B. die *Zählschleife*.

Übung 1.4

Nimm dir deine bisher geschriebenen Programme nochmals vor und ergänze diese um

- Kurzkommentare, die Abschnitte von jeweils wenigen Zeilen erklären und
- einen ausführlichen Kommentarblock zu Beginn, der Filename, Erstellungsdatum, Version, Autor und Zweck des Programmes angibt.

Diese Dokumentation im Programmcode ist so wichtig, daß wir sie künftig in jeder Source verwenden werden.

Ausblick: In der nächsten Übung schreiben wir u.a. ein Programm zu Berechnung von Primzahlen und einen Primzahltest.