Dr. Ole Klein, Dr. Steffen Müthing
IWR, Universität Heidelberg

Abgabe 3. November 2017

Allgemeine Hinweise:

- Die Punkte unter den Aufgaben sind zumindest auf diesem Übungsblatt nicht weiter relevant.
- Das erste Übungsblatt ist dazu gedacht, während der Übung bearbeitet zu werden.
- Während der Übung erhalten Sie Unterstützung von den Übungsleitern und Tutoren. Nutzen Sie diese Möglichkeit und melden Sie sich bei Fragen!

Übung 1 *Einführung in die Kommandozeile*

Benutzen Sie die Kommandozeile, um unter Verwendung der in der Vorlesung vorgestellten Befehle `cd` und `mkdir` in Ihrem Home-Verzeichnis ein Verzeichnis `uebungen` und darin ein Verzeichnis `uebung01` anzulegen.

```
~ $ mkdir uebungen
~ $ cd uebungen
uebungen $ mkdir uebung01
uebungen $ cd uebung01
uebung01 $
```

Wenn Sie nähere Informationen über einen Befehl möchten, schauen Sie sich die Dokumentation des Befehls, die sogenannte *manpage* mit dem Befehl `man BEFEHL` an, z.B. `man mkdir`. Sie können in der Ausgabe mit den Pfeiltasten scrollen und mit der Taste "q" zur Kommandozeile zurückkehren.

Erstellen Sie in diesem Verzeichnis mit einem Texteditor Ihrer Wahl die Datei `helloworld.cc` mit folgendem Inhalt:

```
#include <iostream>

int main(int argc, char** argv)
{
  std::cout << "Hello world!" << std::endl;
  return 0;
}
```

Kompilieren Sie das Programm und führen Sie es aus. Zum Kompilieren verwenden wir hier den GCC-Compiler, die Option "-o NAME" sagt dem Compiler, wie das erzeugte Programm heißen soll.

```
uebung01 $ g++ -o helloworld helloworld.cc
uebung01 $ ./helloworld
Hello world!
uebung01 $
```

Probieren Sie aus, was passiert, wenn Sie die Option "-o" weglassen. Zur Erinnerung: Sie können den Inhalt des aktuellen Verzeichnisses mit dem Befehl `ls` anzeigen.

(5 Punkte)

Übung 2 *Syntaxfehler und Compiler-Meldungen*

Erstellen Sie die Datei `errors.cc` mit folgendem (absichtlich fehlerhaften) Inhalt:

```
#include <iostrean>

int main(int argc, char** argv)
{
  std::cout << "Typing is difficult" << endl;
  int ret = 0;
  return retv;
}
```

Erstellen Sie dann eine Kopie dieser Datei mit dem Befehl `cp errors.cc errors2.cc`.

- Versuchen Sie, das Programm zu kompilieren. Der Compiler wird diverse Fehlermeldungen ausgeben. Beheben Sie den oder die angezeigten Fehler und starten Sie den Compiler erneut. Dies kann unter Umständen dazu führen, dass der Compiler andere Fehler anzeigt. Machen Sie so lange weiter, bis das Programm übersetzt werden kann.
- In der virtuellen Maschine ist als weiterer Compiler `clang++` installiert. Versuchen Sie, das immer noch fehlerhafte Programm in `errors2.cc` mit diesem Compiler zu übersetzen, indem Sie an der Kommandozeile `gcc` durch `clang++` ersetzen. Vergleichen Sie, welcher Compiler die für Sie hilfreicheren Fehlermeldungen generiert. Die beiden Compiler sind im wesentlichen austauschbar, verwenden Sie in Zukunft das Werkzeug, das für Sie am besten funktioniert.

Erstellen Sie die Datei `legalbutwrong.cc` mit folgendem Inhalt:

```
#include <iostream>

int main(int argc, char** argv)
{
  int n = 10;
  // calculate the sum of all numbers from 1 to n
  int i;
  int sum = 0;
  for (int j = 1 ; i <= n ; j = j+1)
  {
    sum = sum + j;
  }
  std::cout << sum << std::endl;
  return 0;
}
```

Kompilieren Sie das Programm und führen Sie es aus. Falls das Programm "hängt", können Sie es mit der Tastenkombination "CTRL-C" beenden.

Das Programm ist zwar syntaktisch korrekt, aber es hat einen Fehler. Der Compiler kann Ihnen oft helfen, solche Probleme aufzuspüren. Hierzu müssen Sie ihn anweisen, Ihnen nicht nur Fehler, sondern auch Warnungen anzuzeigen. Sowohl GCC als auch clang benötigen hierfür die Option "-Wall" (warn all).

Kompilieren Sie das Programm erneut mit der zusätzlichen Option "-Wall" und beheben Sie die vom Compiler gemeldeten Probleme. Das Programm sollte nun die richtige Lösung (55) ausgeben.

(5 Punkte)

Übung 3 *Quadratische Gleichungen*

Schreiben Sie ein Programm, dass von der Kommandozeile die Koeffizienten einer quadratischen Gleichung der Form $ax^2 + bx + c = 0$ abfragt und mit der Mitternachtsformel die beiden Nullstellen ausrechnet und ausgibt. Falls es keine eindeutige Lösung gibt ($a = b = 0$) oder die Lösung komplex ist, soll das Programm das abfangen und eine entsprechende Meldung ausgeben.

Hinweise:

- Um eine Kommazahl von der Kommandozeile einzulesen, verwenden Sie folgenden Codeschnipsel:

```
double v;
std::cout << "a = " << std::flush;
std::cin >> v;
```

- Um die Wurzel aus einer Zahl zu ziehen, gibt es die Funktion `sqrt`:

```
double wurzel2 = std::sqrt(2.0);
```

  Bevor Sie diese Funktion verwenden können, müssen Sie am Anfang des Programms die Mathematik-Bibliothek von C++ mit der Zeile `#include <cmath>` einbinden.

(5 Punkte)

Übung 4 *Fibonacci-Folge*

Jedes Element der Fibonacci-Folge wird durch Addition der beiden vorherigen Folgen-Elemente gebildet, wobei die ersten beiden Elemente durch 0 und 1 gegeben sind:

$$\begin{aligned} f_1 &= 0, \\ f_2 &= 1, \\ f_n &= f_{n-2} + f_{n-1}. \end{aligned}$$

Damit ergibt sich als Beginn der Folge:

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, \ldots$$

- Schreiben Sie ein Programm, das die Fibonacci-Folge bis zum N-ten Term ausgibt, wobei N von der Kommandozeile ausgelesen werden sollten. Verwenden Sie als Datentyp für die Zahlen `int`.
- Probieren Sie Ihr Programm für verschiedene Werte von N aus. Was passiert, wenn Sie N groß werden lassen? Haben Sie eine Erklärung hierfür?

(5 Punkte)