Einführung in die Programmierung, WS 2015/2016
Blatt 6

Simon Hangl, Sebastian Stabinger, Benedikt Hupfauf, Johannes Kessler

2015–11–18

- Abgabe bis spätestens Dienstag 21:59:59 über OLAT (`https://lms.uibk.ac.at/olat/dmz/`).
- Bereiten Sie jede Aufgabe so vor, dass Sie Ihre Lösung im Proseminar präsentieren können!
- Benennen Sie Ihre Abgabe nach folgendem Schema: *Gruppennummer-Nachname*`-blatt`*Übungsblattnummer*`.tar.gz` Wenn Sie also Max Mustermann heißen und Gruppe 1 besuchen, heißt die Datei von Übung 6: `1-mustermann-blatt6.tar.gz`
- Compilieren Sie alle Programme mit den Optionen `-Wall -Werror -std=c99`

Feedback

Nutzen Sie die angebotenen Möglichkeiten, uns Feedback zu geben (eMail, Tutorium, Proseminar). Hier können Sie uns auf Probleme, notwendige Stoffwiederholungen, Unklarheiten, aber auch positive Dinge, die beibehalten werden sollten, hinweisen.

Testen und Dokumentation

Stellen Sie sicher, dass alle Lösungen fehlerfrei kompilieren. Testen Sie Ihre Lösungen ausführlich (z.B. auf falsche Eingaben, falsche Berechnungen, Sonderfälle) und dokumentieren Sie sie. Dies hilft Ihnen bei der Präsentation und uns beim Nachvollziehen Ihrer Entscheidungen. Lesen Sie die Aufgaben *vollständig* durch.

Aufgabe 1 (4 Punkte)

Gegeben ist folgendes Programm:

```
#include <stdio.h>

int main(void) {
  int x = 7;
  printf("zeile 1\n");
  for (int i = 0, j = 3; (i + 5) < x + 10 && j > -50; ++i, j -= 3) {
    printf("zeile 2\n");
    if (j % 4)
      printf("zeile 3\n");
    --i;
  }
  printf("zeile 4\n");
}
```

Schreiben Sie das gegebene Programm auf zwei verschiedene Arten um. Die **erste** Version soll nur eine `while`-Schleife verwenden. Die **zweite** Version soll nur eine `do-while`-Schleife enthalten.

Fragen:

- Lässt sich das Programm optimieren?
- Kann jede `for`-Schleife in eine `while`-Schleife umgewandelt werden?
- Kann jede `while`-Schleife in eine `for`-Schleife umgewandelt werden?

Hinweis: *Abgabe: 1-mustermann-a1_1.c und 1-mustermann-a1_2.c*

Aufgabe 2 (6 Punkte)

Implementieren Sie folgendes Spiel: Der Computer 'überlegt' sich eine Zufallszahl zwischen 1 und 100. Anschließend werden vom Benutzer drei Zahlen abgefragt. Ist die gesuchte Zahl (die vorher generierte Zufallszahl) dabei, gewinnt der Spieler und das Programm endet. Ist die richtige Zahl nicht dabei, erfährt der Spieler ob die Mehrheit der Zahlen größer oder kleiner als die gesuchte Zahl ist. Im Anschluss werden erneut 3 Zahlen vom Benutzer abgefragt. Dies wiederholt sich solange der Spieler die gesuchte Zahl nicht erraten hat.

```
Beispiel:
Computer erzeugt die Zahl 77
Der Benutzer gibt 99 55 1 ein
Der Computer gibt 'Die gesuchte Zahl ist größer' aus, da zwei Zahlen
kleiner als die gesuchte Zahl sind, aber nur eine größer.
Der Benutzer gibt 44 78 95 ein.
Der Computer gibt 'Die gesuchte Zahl ist kleiner' aus, da zwei Zahlen
größer als die gesuchte Zahl sind, aber nur eine kleiner.
Der Benutzer gibt 37 77 und 17 ein.
Der Computer gibt 'Sie haben in 3 Zügen gewonnen!' aus und beendet
das Programm.
```

Hinweis: Sie können mit dem Befehl `srand(time(NULL));` einen Zufallszahlengenerator initilisieren und anschließend mit `(rand() % 100) + 1;` eine Zufallszahl zwischen 1 und 100 erzeugen. Dazu wird der Header `stdlib.h` benötigt.

Hinweis: *Abgabe: 1-mustermann-a2.*c