Universität Konstanz
Informatik & Informationswissenschaft
Prof. Dr. Marcel Waldvogel / Klaus Herberth

Programmierkurs 3
WS 2016/2017

1. Übungsblatt

(Ausgabe: 27. Oktober 2016 — Abgabe bis: 3. November 2016, 04:00)

Folgende Regeln gelten für *alle* Abgaben:

- Abgabe ist jeweils um 4 Uhr morgens am Tag der nächsten Übung.
- Verspätete Abgaben werden nicht gewertet. Es zählt der Repo-Zustand auf den wir zum Abgabezeitpunkt zugreifen können.[1]
- Damit die Abgabe gültig ist, müssen alle Punkte von Aufgabe 4 auf diesem Blatt erfüllt sein.
- Es wird immer in Zweierteams abgegeben.
- Auf allen zu bewerteten Dokumenten müssen beide Namen stehen.
- Textaufgaben werden als Markdown-Dokument mit dem Namen `abgabe.md` abgegeben. Programmieraufgaben werden als Quelltextdokument mit dem angegebenem Namen abgegeben.
- Grafiken müssen immer in das Markdown-Dokument eingebettet sein.
- Ankündigungen werden als Issues im Projekt-Repository geposted. (Wir empfehlen dazu die Aktivierung einer automatischen Benachrichtigung.)

Aufgabe 1: Ankommen & Einrichten (35 Punkte)

In dieser ersten Aufgabe installieren Sie bitte *Ubuntu 16.04* in einer virtuellen Maschine, kompilieren Minix und starten damit ihren BeagleBone Black nach der bereitgestellten Anleitung. Tragen Sie Ihre Namen in das Startbanner ein und fügen Sie einen Screenshot davon in die Abgabe ein. Erklären Sie zudem die Funktion aller in der Anleitung verwendeten Kommandos und deren Parameter.

Aufgabe 2: Snapshots (2 Punkte)

Erstellen Sie einen Snapshot ihrer virtuellen Maschine und fügen Sie davon einen Screenshot in das Abgabe-Markdown ein.

[1] Ob etwas im Repo auf `git.uni.kn` ist, seht ihr am besten im Webinterface. Wenn ihr uns Zugriffsrechte gewährt habt, können wir auf den Stand zugreifen.

Aufgabe 3: Manpages (5 Punkte)

Manpages sind Hilfe- und Dokumentationsseiten unter UNIX und verwandten Betriebssystemen, z.B. Linux.
Lesen Sie die Manpage vom Kommando `man`, indem Sie das Kommando `man man` ausführen. Geben Sie dabei Antworten auf folgende Fragen:

1. Wie kann man eine Manpage verlassen? (1)
2. Wie kann man in einer Manpage nach Text suchen? (1)
3. Aus welchen Abschnitten kann eine Manpage bestehen? (1)
4. Was wird normalerweise im Abschnitt `SYNOPSIS` angegeben? (1)
5. Wie sucht man nach Manpages, in den ein bestimmtes Schlüsselwort vorkommt? (1)

Aufgabe 4: Fork Repository (7 Punkte)

Forken Sie das Repository `https://git.uni-konstanz.de/info3/bspk` über das Webinterface von Gitlab, clonen Sie Ihren Fork via SSH (entweder in ihrer VM oder lokalen Rechner[2]) und fügen Sie das original-Repository als *Upstream Remote* hinzu. Welche Kommandos sind hierfür notwendig und wie können Sie Upstream-Änderungen in Ihren Fork einspielen? Checken Sie eine Textdatei mit den Namen der Gruppenmitglieder ein und pushen Sie es in Ihren Fork. Geben Sie in der Weboberfläche von Gitlab der Gruppe `info3` maximale Berechtigungen (master) und stellen Sie die Sichtbarkeit ihres Forks auf privat. (Anderenfalls ist Ihre Abgabe nicht möglich bzw. ungültig.)

Hinweis: Um pushen zu können, müssen Sie SSH nutzen. Dies bedeutet, dass Sie ihren Fork mit der SSH-URL geklont haben (beginnt mit `git@git.uni-konstanz.de`), Sie im Besitz eines privaten SSH-Schlüssels sind und den öffentlichen Teil in Gitlab hinterlegt haben. Eine Anleitung, inklusive Key-Generierung unter Linux (Ihre VM), finden Sie zum Beispiel unter `https://docs.gitlab.com/ee/gitlab-basics/create-your-ssh-keys.html`.

Aufgabe 5: Common Commands (6 Punkte)

Benutzen Sie die entsprechenden Manpages und beschreiben Sie kurz was die folgenden Kommandos machen:

1. `cat`, `pwd`*, `ls`, `cd`*, `mv`, `rm`, `mkdir`; (je $\frac{1}{2}$ Punkt)
2. `cal, date, time, watch`; (je $\frac{1}{2}$ Punkt)
3. `find`. ($\frac{1}{2}$ Punkt)

*dies ist ein in die Shell eingebautes Kommando. Seine Beschreibung finden Sie in `bash`(1) im Abschnitt `SHELL BUILTIN COMMANDS`.
Bonuspunkt: Wieso sind `cd` und `pwd` im Gegensatz zu den anderen Kommandos in der Shell eingebaut?

[2]Wir empfehlen das Forken auf ein Linuxsystem.

Aufgabe 6: Hello YOUR_NAME (5 Punkte)

Schreiben und kompilieren Sie ein C-Programm (in Ihrer virtuellen Maschine), dass Sie nach Ihrem Namen fragt und Sie dann persönlich begrüßt. Fügen Sie bitte den Quelltext in einer Datei namens `hello.c` Ihrer Abgabe hinzu. Nutzen Sie bitte die entsprechenden Flags für `gcc` die in der Vorlesung vorgestellt wurden und beachten Sie die Regeln zu Punktabzug bzw. Bonuspunkten.

```
$ gcc -std=c99 -g -Wall -Wextra -Wpedantic -Wbad-function-cast \
-Wconversion -Wwrite-strings -Wstrict-prototypes hello.c
$ checkpatch.pl --no-tree --no-signoff -f --ignore NEW_TYPEDEFS,\
AVOID_EXTERNS,GLOBAL_INITIALISERS,BLOCK_COMMENT_STYLE hello.c
```