J. Gutenberg-Universität Mainz
Prof. Dr. Reinhold Kröger
Fabian Meyer M.Sc.

Betriebssysteme
WS 2012/13

Übungsblatt 1
Praktische Übungen

Aufgabe 1.1 (UNIX-Kommandos):

Wiederholen Sie den Umgang mit den allgemeinen UNIX-Kommandos. Im weiteren wird die Kenntnis des praktischen Umgangs mit dem UNIX-System auf Kommandoebene als bekannt vorausgesetzt. Sie können sich z.B. orientieren an:

- Online-Dokumentationsprojekt SelfLinux (http://www.selflinux.org/selflinux/)
- Single Unix Specification (siehe Materialliste oder http://www.opengroup.org)

Aufgabe 1.2 (Programmentwicklung unter UNIX):

(a) Wissen Sie noch, wie Programmentwicklung unter UNIX geschieht? Wenn Sie in dieser Hinsicht noch oder wieder unsicher sein sollten, können Sie die Notizen "Übersicht zur Programmentwicklung unter UNIX" durcharbeiten. Diese enthalten auch Hinweise bezüglich der Vorgehensweise zur Bearbeitung der weiteren Teilaufgaben. Die Notizen sind als PDF-Datei `C_dev_kz.pdf` über die Materialienseite zur Vorlesung erhältlich. Einzustellende Optionen für die Werkzeuge zur Programmerstellung erfahren Sie wie üblich mit dem `man`-Kommando.

(b) Legen Sie sich in Ihrem home-Verzeichnis ein Unterverzeichnis `uebung_1` an und kopieren Sie in dieses Verzeichnis die bereitgestellte Datei `gesamt.c`.

(c) Verstehen Sie das Programm. Übersetzen Sie es und führen Sie alle Wahlmöglichkeiten aus. Aus funktionaler Sicht ist das Programm sehr einfach. Betrachten Sie zwischendurch den C-Quellcode und ergründen Sie in Zweifelsfällen das Laufzeitverhalten aus dem Quellcode heraus und umgekehrt.

(d) Erzeugen Sie sich im Verzeichnis `uebung_1` ein Unterverzeichnis `zerlegt`, darin wiederum ein Unterverzeichnis `include`, und zerlegen Sie das gegebene Programm in folgende Module:

(1) einen Modul Strichrechnung, der die Strich-Operationen add und sub enthält. Der Modul soll gebildet werden durch eine Datei `strich.c` und eine zugehörige Header-Datei `strich.h`, die die Prototypen der Funktionen enthält und im Unterverzeichnis `include` abgelegt wird.

(2) einen analogen Modul Punktrechnung, der die Punkt-Operationen mult und div enthält und durch die Dateien `punkt.c` und `punkt.h` (ebenfalls im Verzeichnis `include`) gebildet wird.

(3) einen analogen Modul für die Restklassenoperationen mit den Dateien `rest_op.c` und `rest_op.h`.

(4) das Hauptprogramm, das neben der Funktion `main()` die definierten Hilfsfunktionen enthält und aus den Dateien `hpt.c` und `hpt.h` gebildet wird. Die Header-Datei soll dabei Konstantendefinitionen für das Hauptprogramm aufnehmen.

Erzeugen Sie das lauffähige Programm `hpt1` aus all diesen Komponenten. (Es sollte sich selbstverständlich genauso verhalten wie das ursprüngliche Programm `gesamt.c`).

(e) Führen Sie im Verzeichnis `zerlegt` das Unterverzeichnis `lib` ein. Erstellen Sie in diesem Verzeichnis eine Objekt-Bibliothek `libsp.a`, die die Module Strichrechnung und Punktrechnung enthält (noch nicht die Restklassenoperationen). Erzeugen Sie nun das Hauptprogramm mit Namen `hpt2` unter Verwendung der Bibliothek.

(f) Erzeugen Sie für den Modul `rest_op.c` der Restklassenoperationen eine Assembler-Datei `rest_op.s` und assemblieren Sie diese, so dass Sie anschließend unter Verwendung der erzeugten Objektdatei und der Objektbibliothek eine neue Version `hpt3` erzeugen können. Schauen Sie ruhig mal in die Assembler-Datei hinein! (Wenn Sie wollen, können Sie nun Ihre Bibliothek um die Restklassenoperationen erweitern.)

(g) Erzeugen Sie unter Verwendung des Präprozessors das expandierte Hauptprogramm und unter Verwendung dieser Datei und der Objektbibliothek eine neue Version `hpt4`. (Was können Sie in der expandierten Datei beobachten?)

Aufgabe 1.3 (Programmierstil):

Im weiteren des Praktikums wird davon ausgegangen, dass Sie sich einen auch für andere Personen lesbaren Programmierstil angewöhnt haben. Falls Sie noch unsicher sein sollten, beachten Sie z.B. die kleinen Style Guides für C/C++, die unter Materialien ebenfalls bereit gestellt werden.