Einführung in die Programmierung
WS 2015/2016
Blatt 11

Simon Hangl, Sebastian Stabinger, Benedikt Hupfauf, Johannes Kessler

2015–01–20

- 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 11: `1-mustermann-blatt11.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 (2 Punkte)

In dieser Aufgabe soll eine kleine Adressdatenbank implementiert werden. Ein Datensatz enthält **Vorname, Nachname, Adresse und Alter** einer Person. Ein Datensatz soll jeweils in einem `struct` gespeichert werden.

In der Aufgabenstellung finden Sie eine Headerdatei namens `address.h` in welcher Signaturen für zwei Funktionen definiert sind welche zum Einlesen und Ausgeben eines solchen Adressdatensatzes verwendet werden können. **Definieren Sie** zuerst eine sinnvolle Struktur in der Headerdatei und **implementieren Sie** die beiden vorgegebenen Funktionen in einer Datei `address.c`.

Implementieren Sie anschließend eine Datei `menu.c` welche `address.h` und `address.c` verwendet um ein Array mit Adressdatensätzen zu füllen. Geben Sie dem Benutzer in einem Menü die Möglichkeit einen *neuen Datensatz* hinzuzufügen, alle vorhandenen *Datensätze auszugeben* und das Programm zu *beenden*.

Sie können für die Implementierung ein Array fixer Größe verwenden.

Frage: Sie haben jetzt mehr als eine Datei. Wie muss dieses Programm nun compiliert werden? Muss die Headerdatei beim compilieren angegeben werden? Falls nicht: Warum nicht?

Hinweis: *Abgabe: address.h*
Hinweis: *Abgabe: address.c*
Hinweis: *Abgabe: menu.c*
Hinweis: *Abgabe: answers.txt*

Aufgabe 2 (4 Punkte)

Verwenden Sie die Lösung der vorigen Aufgabe und implementieren Sie eine weitere Funktion `sort`, welche ein eingelesenes Array mittels Quicksort sortiert. Verwenden Sie dazu Libraryfunktionen wie `qsort` oder `qsort_r` (siehe letztes Übungsblatt). Implementieren Sie entsprechende Vergleichsfunktionen für folgende Sortierungen:

- Aufsteigend nach Vornamen
- Absteigend nach Nachnamen
- Aufsteigend nach der Länge des Vornamens. Wenn zwei Vornamen dieselbe Länge haben, wird auch die Länge des Nachnamens verglichen (`strlen` könnte hilfreich sein)

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