Institut für Informatik
Prof. Dr. Stefan Conrad

Sommersemester 2021
Übung 3

Hardwarenahe Programmierung

Gruppe 11 (Thomas)

In den bisherigen Übungen haben Sie Nutzereingaben zur Laufzeit des Programms über Tastatureingaben implementiert. In dieser Übung schreiben Sie C-Programme, die ihren Input als Kommandozeilen-Parameter und -Optionen erhalten, und die Ausgabedaten ebenfalls in Dateien schreiben können.

Hinweis: *Zur Vereinfachung der Aufgaben müssen Sie unrealistische oder falsche Eingaben nicht abfangen, wenn nicht explizit gefordert. Zum Beispiel müssen Sie sich keine Gedanken darüber machen, was passiert, wenn eine Zahl kleiner als 2^{31} oder größer als $2^{31} - 1$ oder sogar ein Text eingegeben wurde, wo eine Jahreszahl erwartet wird.*
Denken Sie wie immer daran, Ihre Lösungen im ILIAS hochzuladen und den Test im ILIAS zu absolvieren!

Aufgabe 1 *Kleiner Texteditor (**Pflichtaufgabe**)*

In dieser Aufgabe sollen Sie einen Texteditor implementieren, der einen Text und Anweisungen über Kommandozeilenparameter entgegennimmt und modifizierten Text ausgibt.

(a) Erstellen Sie hierzu eine Datei `texteditor.c`. Das Programm soll drei Optionen behandeln können:

- `-k`: Konvertiert alle Großbuchstaben zu Kleinbuchstaben.
- `-t <Zwischentext>`: Gibt den spezifizierten Text zwischen jedem Buchstaben des Eingabetextes aus.
- `-h`: Gibt eine Hilfestellung zur Benutzung des Texteditors aus und beendet dann das Programm. In der Ausgabe muss das Wort *Hilfe* enthalten sein.

Der verbleibende Parameter (ohne `-`) ist der zu verarbeitende Text.

Bei folgendem Beispielaufruf wird der Text zu Kleinbuchstaben konvertiert und ein Komma zwischen aufeinanderfolgenden Buchstaben eingefügt:

```
> ./texteditor -k -t ',' 'ABC'
a,b,c
```

Ein weiteres Beispiel:

```
> ./texteditor Nnnnnnnnnnn -t 'a' && printf 'a Batman!'
Nanananananananananananana Batman!
```

(b) Bei zu vielen oder zu wenigen Parametern (fehlendem Eingabetext oder mehrere Eingabetexte) sowie unbekannten Optionen (zum Beispiel `-x`) soll sich das Programm beenden, den Wert 1 zurückgeben und eine Fehlermeldung ausgeben, die das Wort *Fehler* beinhaltet.

Falls sowohl falsche Parameter als auch die Option `-h` übergeben wurde dürfen Sie sich aussuchen, ob Sie eine Fehlermeldung oder Hilfe ausgeben.

Sie dürfen annehmen, dass jede Option höchstens einmal übergeben wird.

(c) Verwenden Sie das Skript `test.sh`, um ihr Programm zu überprüfen.

Aufgabe 2 *Studierende (**Pflichtaufgabe**)*

In dieser Aufgabe erhalten Sie Listen von Studierenden über die Standardeingabe, die Sie nach Matrikelnummer, Studiengang, Semester und Rückmeldestatus filtern und zeilenweise ausgeben sollen.

- Eine Liste enthält vier Spalten. In der ersten Spalte steht die Matrikelnummer, in der zweiten das Semester, in der dritten der Studiengang und in der vierten, ob die Semestergebühr entrichtet wurde.

  ```
  9562    6    Mathematik    ja
  7598    4    Physik        nein
  5580    7    Informatik    ja
  ```

  Es darf angenommen werden, dass keine Zeile mehr als 160 Zeichen enthält.

- Ihr Programm soll in der Datei `studierende.c` implementiert werden und folgende Kommandozeilenparameter verarbeiten können:

`-g zahl`	Betrachtet nur Studierende mit einer Matrikelnummer größer als `zahl`.
`-k zahl`	Betrachtet nur Studierende, die kürzer als `zahl` Semester studieren.
`-s string`	Betrachtet nur Studierende im Studiengang `string`.
`dateiname`	Wenn ein anderer Parameter als die vorherigen drei übergeben wird, dann soll angenommen werden, dass dies ein Dateiname ist. In diesem Fall soll die Ausgabe in diese Datei anstatt auf `stdout` ausgegeben werden.

  Wenn mehrere Optionen übergeben werden, dann sollen nur Studenten betrachtet werden, die alle Eigenschaften erfüllen. Es darf angenommen werden, dass jede Option höchstens einmal an das Programm übergeben wird.

- Verwenden Sie `test.sh` zum testen Ihres Programms. Dieses Test-Skript ruft Ihr Programm mit verschiedenen Parametern auf und vergleicht die Ausgabe mit den erwarteten Ausgaben, die wir Ihnen ebenfalls zur Verfügung stellen.

Beispielaufrufe:

`./studierende < studierende.txt`

Gibt alle Studierenden aus.

`./studierende < studierende.txt > ausgabe.txt`

Schreibt alle Studierende in die Datei `ausgabe.txt`.

`./studierende ausgabe.txt < studierende.txt`

Schreibt ebenfalls alle Studierende in die Datei `ausgabe.txt`.

`./studierende -g 2000 < studierende.txt`

Gibt Studierende mit Matrikelnummer größer als 2000 aus.

`./studierende -s Informatik < studierende.txt`

Gibt alle Informatik-Studenten aus.

`./studierende -k 5 -s Mathematik < studierende.txt`

Gibt alle Mathematik-Studenten aus, die weniger als 5 Semester studieren.