Institut für Informatik
Prof. Dr. Stefan Conrad

Sommersemester 2021
Übung 3

Hardwarenahe Programmierung
Gruppe 17 (Florian)

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 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, den Test im ILIAS zu absolvieren und das zugehörige Kapitel im Lernmodul durchzuarbeiten!

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

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

Eine Zeile beginnt am Anfang eines Textes oder nach einem Zeilenumbruch.

Falls die letzte Zeile leer ist, also ein Zeilenumbruch direkt vom Nullterminator gefolgt wird, steht Ihnen frei, ob Sie vor dieser Zeile auch eine Zeilennummer ausgeben oder nicht. Dieser Sonderfall wird nicht überprüft.

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

- `-s <Zahl>`: Die Zeilennummern starten bei dieser Zahl (sonst bei 1).
- `-t <Zwischentext>`: Gibt den spezifizierten Text zwischen jeder Zeilennummer und der entsprechenden Zeile 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 werden Zeilennummern beginnend bei 9 vergeben und zwischen der Zeilennummer und der entsprechenden Zeile noch der Text „ - “ausgegeben.

Aufruf. Hier wird ein mehrzeiliger String als Kommandozeilenparameter übergeben:

```
./texteditor -s 9 -t ' - ' 'Eins
Zwei
Drei'
```

Ausgabe:

```
9 - Eins
10 - Zwei
11 - Drei
```

(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 die Option `-h` als auch falsche Parameter übergeben wurden 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.

Hinweis: Diese Aufgabe kann ohne zusätzliche char-Arrays gelöst werden. Dies ist sogar ratsam, da lange Texte sonst den Stack sprengen könnten, der nur eine begrenzte Größe hat. Die Arrays in `argv` zu modifizieren ist zudem keine gute Idee, da diese ebenfalls eine begrenze Größe haben.

Aufgabe 2 *Lebensmittel (**Pflichtaufgabe**)*

In dieser Aufgabe verwalten Sie einen Supermarkt. Sie erhalten Listen von Lebensmitteln über die Standardeingabe, die Sie nach Bezeichnung, Preis, Kategorie und Mindesthaltbarkeit filtern und zeilenweise ausgeben sollen.

- Eine Liste enthält vier Spalten. In der ersten Spalte steht die Bezeichnung, in der zweiten der Preis (mit zwei Nachkommastellen), in der dritten die Kategorie und in der vierten die Mindesthaltbarkeit in Tagen:

  ```
  Kirschkuchen 4.00  Backware  5
  Kirschsaft   1.50  Getränk  30
  Sauerkirsche 1.50  Obst      8
  ```

  Es darf angenommen werden, dass keine Zeile mehr als 135 Zeichen enthält.

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

`-p <Kommazahl>`	Betrachtet keine Lebensmittel mit höherem Preis.
`-d <Ganzzahl>`	Betrachtet keine Lebensmittel mit niedrigerem Haltbarkeitsdatum.
`-b <Bezeichnung>`	Betrachtet keine Lebensmittel mit anderer Bezeichnung.
`Dateiname`	Wenn ein anderer Parameter als die vorherigen drei übergeben wird, dann soll angenommen werden, dass es sich hierbei um einen Dateiname handelt. In diesem Fall soll die Ausgabe in diese Datei anstatt auf `stdout` ausgegeben werden.

  Wenn mehrere Optionen übergeben werden, dann sollen nur Lebensmittel 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:

`./lebensmittel < lebensmittel.txt`

Gibt alle Lebensmittel aus.

`./lebensmittel < lebensmittel.txt > ausgabe.txt`

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

`./lebensmittel ausgabe.txt < lebensmittel.txt`

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

`./lebensmittel -p 1.50 < lebensmittel.txt`

Gibt alle Lebensmittel aus, die `1.50` oder weniger kosten.

`./lebensmittel -b Ei < lebensmittel.txt`

Gibt alle Eier aus.

`./lebensmittel -d 4 -b Apfel < lebensmittel.txt`

Gibt alle Äpfel aus, die mindestens 4 Tage haltbar sind.