Dipl.Phys. Gerald Kempfer

Beuth Hochschule für Technik Berlin
- University of Applied Sciences -
Fachbereich VI – Informatik und Medien
Studiengang Technische Informatik Bachelor

Lehrveranstaltung "Informatik II für TI-Bachelor"
Übungsblatt 1

Hinweise:

Dieses Übungsblatt ist zur Zulassung zu der Klausur erfolgreich zu bearbeiten ("*Erfolgreich*" bedeutet: Keine Programmabstürze bzw. Endlosschleifen, Aufgabenstellung einschl. der Nebenbedingungen müssen eingehalten sowie Kommentierung und Einrückung korrekt sein!).

Die Aufgaben werden überwiegend in den Übungszeiten bearbeitet. Allerdings genügt die Zeit hierfür unter Umständen nicht, so dass Sie auch außerhalb dieser Zeiten die Aufgaben bearbeiten müssen. Der Abgabetermin für diese Aufgabe ist **spätestens** der **21. Oktober 2016 (Gruppe 1) bzw. 28. Oktober 2016 (Gruppe 2)**.

Nutzen Sie die Übungen auch, um ggf. Fragen, die sich in den Vorlesungen ergeben haben, anzusprechen.

Aufgabe:

Ziel der ersten Übung ist das Kennenlernen der Arbeitsmittel, die in diesem Semester benötigt werden: PCs des LIS-Labors und die Entwicklerumgebung Code::Blocks.

Im ersten Teil werden wir den Zugang zu den PCs einrichten.

Dann werden wir gemeinsam mit der Entwicklerumgebung Code::Blocks (kann kostenlos von www.codeblocks.org für Windows, Linux und Mac heruntergeladen werden) ein kleines Projekt erstellen, dieses compilieren, linken und starten. Ferner werden wir das Debuggen ausprobieren: Setzen von Breakpoints, schrittweise Ausführung des Programms, Anzeigen von Variableninhalten (Watches).

Im zweiten Teil der Übung werden Sie in Dreiergruppen die eigentliche erste Übung bearbeiten. Dabei sollen alle Bezeichnungen in Englisch geschrieben werden.

Erstellen Sie als erstes die Headerdatei `datastructure.h`. In dieser Datei sollen zwei Datenstrukturen mittels `typedef` deklariert werden:
- `TDate`: beinhaltet drei Zahlen für Tag, Monat und Jahr
- `TTime`: beinhaltet drei Zahlen für Stunde, Minute und Sekunde.

Schreiben Sie dann die fehlenden Funktionen zum vorgegebenen Hauptprogramm. Diese Funktionen sollen in einem eigenen Modul (z.B. `datetime.c`) untergebracht werden, da diese in den weiteren Übungsaufgaben wieder benötigt werden.

Die Funktion `isLeapYear` (soll von der nächsten Funktion aufgerufen werden) soll als Funktionsergebnis eine ganze Zahl zurückgeben. Diese

Zahl soll als Wahrheitswert angeben, ob das angegebene Jahr (Parameter) ein Schaltjahr ist.

Die Funktion `isDateValid` (soll von der nächsten Funktion aufgerufen werden) soll als Funktionsergebnis eine ganze Zahl zurückgeben. Diese Zahl soll als Wahrheitswert angeben, ob das angegebene Datum (Parameter Struktur `TDate` mit Tag, Monat und Jahr) ein gültiges Datum ist. Dabei sollen auch die Schaltjahre berücksichtigt werden.

Die Funktion `getDateFromString` (wird vom Hauptprogramm aufgerufen) soll als Funktionsergebnis eine ganze Zahl zurückgeben. Diese Zahl soll als Wahrheitswert angeben, ob in der angegebenen Zeichenkette (1. Parameter) ein Datum enthalten ist. Dieses Datum soll in Tag, Monat und Jahr (als Zahlen) zerlegt – sozusagen geparst – werden; dabei sind die Zahlen durch Punkte voneinander getrennt; die Punkte müssen nicht unbedingt an den Positionen 3 und 6 stehen (siehe Beispielausgabe)! Natürlich sollen diese drei Zahlen als Datum geprüft werden, ob sie ein gültiges Datum ergeben. Wenn ja, sollen diese drei Zahlen in die übergebene Datumsstruktur (2. Parameter: Zeiger auf `TDate`) gespeichert werden.

Analog dazu soll auch eine Eingabe einer Uhrzeit möglich sein. Dazu werden folgende Funktionen benötigt:
- `isTimeValid`: prüft, ob die in der Zeitstruktur (Datentyp `TTime`) angegebene Uhrzeit gültig ist;
- `getTimeFromString`: liest aus einer Zeichenkette die Stunden, Minuten und Sekunden und speichert diese – sofern sie eine gültige Uhrzeit bilden – in der Zeitstruktur.

Es ist komplett analog zu der Datumseingabe vorzugehen – mit Ausnahme, dass hier keine Schaltstunden oder Ähnliches existieren und dass die Gültigkeitsprüfung einfacher ist, da hier jede Stunde die gleiche Anzahl von Minuten hat. Ferner werden zur Trennung der Zahlen Doppelpunkte anstelle von Punkten verwendet.

Ferner soll wieder wie im vorigen Semester ein Modul namens `tools.c` erstellt werden mit Hilfsfunktionen wie
- `clearBuffer()`
- `waitForEnter()`
- `clearScreen()` (hier wird die Funktion `system` aus der Headerdatei `stdlib.h` verwendet, z.B. `system("CLS");`)
- `askAgain()`

Dieses Modul wird im Laufe dieses Semesters immer wieder benötigt und wird noch erweitert werden.
Unter Linux kann auch wieder die Headerdatei `escapesequenzen.h` verwendet werden (dies funktioniert leider nicht unter Windows).

Die Headerdatei `string.h` darf nicht verwendet werden!

Kommentieren Sie das Programm. Dazu gehört auch ein Modulheader und zu jeder Funktion ein Funktionsheader (siehe Skript “Grundlagen der Informatik“ Kapitel 5.3 und 5.4)! Achten Sie auch auf Ihre Programmstruktur (Einrückungen, Leerzeichen und -zeilen).

Bei der Abgabe soll die Funktion `getDateFromString` von Ihnen schrittweise vorgeführt und dabei alle Variableninhalte angezeigt werden.

Beispielausgabe:

Die Bildschirmausgabe soll folgendermaßen aussehen:

```
Geben Sie bitte ein gueltiges Datum ein: 31.02.2016
Das eingegebene Datum 31.02.2016 ist ungueltig!
Geben Sie bitte eine gueltige Uhrzeit ein: 25:59:59
Die eingegebene Uhrzeit ist ungueltig!
Moechten Sie noch einmal (j/n) ? jau
Geben Sie bitte ein gueltiges Datum ein: 003.0010.2016
Das Datum 03.10.2016 ist gueltig!
Geben Sie bitte eine gueltige Uhrzeit ein: 8:0:0
Die Uhrzeit 08:00:00 ist gueltig!
Moechten Sie noch einmal (j/n) ? nö
```

Hauptprogramm:

```
#include <stdio.h>
#include "datastructure.h"
#include "datetime.h"
#include "tools.h"

int main()
{
   TDate Date;
   TTime Time;
   char Input[20];

   do
   {
      clearScreen();
      printf("Geben Sie bitte ein gueltiges Datum ein: ");
      *Input = '\0';
      scanf("%19[^\n]", Input);
      clearBuffer();
      if (*Input)
         if (getDateFromString(Input, &Date))
            printf("Das Datum %02i.%02i.%04i ist gueltig!\n",
                   Date.Day, Date.Month, Date.Year);
         else
            printf("Das eingegebene Datum %s ist ungueltig!\n", Input);
      else
         printf("Sie haben nichts eingegeben!\n");

      printf("Geben Sie bitte eine gueltige Uhrzeit ein: ");
      *Input = '\0';
      scanf("%19[^\n]", Input);
      clearBuffer();
      if (*Input)
         if (getTimeFromString(Input, &Time))
            printf("Die Uhrzeit %02i:%02i:%02i ist gueltig!\n",
                   Time.Hour, Time.Minute, Time.Second);
         else
            printf("Die eingegebene Uhrzeit ist ungueltig!\n");
      else
         printf("Sie haben nichts eingegeben!\n");
   } while (askYesOrNo("Moechten Sie noch einmal (j/n) ? "));

   return 0;
}
```