3. Praktische Übung in "Programmieren": Messdatei

Eine Umweltmessstation erfasst in **10-Minuten-Intervallen** die Werte der **Radioaktivität** (Ortsdosisleistung in µSv/h) und die **Temperatur** (in °C). Die sich i.a. über **mehrere Tage** erstreckenden Messdaten sind in einer Textdatei (**Messdatei**) abgelegt, deren Aufbau der folgende Ausschnitt des Beginns einer Datei zeigt:

```
 Messzeit       uSv/h   °C      Startdatum: 7.2.2009
 8:50:00        0.101   23.7
 9:00:00        0.102   23.5
 9:10:00        0.102   23.7
 9:20:00        0.102   23.6
 9:30:00        0.098   23.5
           .........
```

Sie haben die Aufgabe, die Messdaten weiter zu verarbeiten. Dazu soll Ihr Programm in einer Schleife das folgende **Auswahl-Menü** anbieten:

```
Name der Messdatei: I070209.dat

(1) Inhalt der Messdatei
(2) Extrahiere 24-Std-Binaerdatei
(3) Anzeige der 24-Std-Binaerdatei
(E) EXIT
```

(1) zeigt den Inhalt der aktuellen Messdatei zeilenweise an, wobei nach jeweils 20 Zeilen die Ausgabe anhält und nach Tastatureingabe wahlweise fortgesetzt oder abgebrochen werden kann.

(2) erstellt eine Binärdatei `"*.bin"`, deren Namen sich lediglich in der Extension vom Namen der Messdatei unterscheidet (für die Messdatei `"I070209.dat"` also `"I070209.bin"`). Falls der Messdateiname keine Extension besitzt, ist `".bin"` an den Namen anzuhängen.
Der Inhalt der Binärdatei sind 24-Stunden-Mittelwerte der Ortsdosisleistung und der Temperatur, sowie die korrespondierenden Streuungsmaße (Standardabweichung, Formeln s.unter "Vorgaben zur Lösung").
Ein – 24 Std umfassendes – **Auswerte-Intervall startet zur ersten Mitternacht** nach Beginn der Messreihe Messdaten vor diesem Zeitpunkt und nach dem letzten abgeschlossenen Intervall sind zu verwerfen.
Es kann davon ausgegangen werden, dass in einem abgeschlossenen Intervall keine Messdaten "fehlen", d.h. pro 24 Stunden immer 144 Messwerte vorhanden sind.

Die in der Binärdatei abzulegende Information besteht aus Datensätzen, die durch den folgenden **Structure-Typ** beschrieben werden:

```
typedef struct
{ double owert;     // Ortsdosisleistung, 24-Std-Mittelwert
  double osig;      // Ortsdosisleistung, Standardabweichung
  double twert;     // Temperatur, 24-Std-Mittelwert
  double tsig;      // Temperatur, Standardabweichung
} MWTyp;
```

(3) gibt den Inhalt der erstellten 24-Std-Binärdatei geeignet am Bildschirm aus (pro Zeile einen Datensatz).

(E) beendet das Programm

Vorgaben zur Lösung

- Der **Name** der **Messdatei** ist dem Programm als **Parameter** zu übergeben. Falls **kein Programmparameter** übergeben wird, ist der Name von der **Standardeingabe einzulesen**.
  Zwei geeignete Messdateien werden zu Verfügung gestellt.

- Es ist durch **Aufruf** einer geeignet **zu definierenden Funktion** zu **überprüfen**, ob die durch den Namen referierte **Datei existiert**. Falls die Datei **nicht existiert** ist das Programm mit einer entsprechenden **Meldung** zu beenden, **andernfalls** ist in einer **Schleife** das o.a. **Auswahl-Menü** auszugeben.

weiter Seite 2

Vorgaben zur Lösung, Forts.

- Jeder der **Menüpunkte (1) bis (3)** ist durch eine **eigene Funktion** zu realisieren (Menü-Funktionen).
  Jede dieser Funktionen besitzt einen **`char*`** (`char`–Pointer) als **einzigen Parameter** und gibt **keinen Funktionswert** zurück. Als aktueller Parameter wird allen drei Funktionen der Messdatei-Name übergeben.
  Treten beim **Öffnen der Datei Fehler** auf, geben die Funktionen eine entsprechende **Fehlermeldung** aus.
  Die **Funktionsnamen** sind von Ihnen jeweils **geeignet zu wählen**.
  Diese Funktionen sind in einer **eigenen** – von `main()` getrennten – **Quelldatei** zu implementieren.

- In der **`main()`**–Funktion sind die **Pointer** auf die **Menü-Funktionen** in einem geeigneten **Funktions-Pointer-Array** zusammenzufassen.
  Nach Ermittlung des Messdatei-Namens und der erfolgreichen Überprüfung auf Existenz dieser Datei gibt `main()` in einer **Schleife** das Auswahl-Menü in die Standard-Ausgabe aus, liest die jeweilige Menü-Auswahl von der Standard-Eingabe ein und ruft die der getroffenen Auswahl entsprechende Menü-Funktion mittels **Indizierung des Funktionspointer-Arrays** auf (**kein Aufruf** der Menü-Funktionen **über** ihren **Namen!**).
  Eine **ungültige Menü-Auswahl** wird **ignoriert**.

- Für die Ermittlung des **Mittelwerts** und der **Standardabweichung** sind die **folgenden Formeln** anzuwenden:

$$\bar{x} = \frac{1}{n} * \sum_{i=1}^{n} x_i \qquad \text{Mittelwert}$$

$$s = \sqrt{\frac{1}{n} * \sum_{i=1}^{n} (x_i - \bar{x})^2} = \frac{1}{n} * \sqrt{n * \sum_{i=1}^{n} x_i^{\,2} - (\sum_{i=1}^{n} x_i)^2} \qquad \text{Standardabweichung}$$

  Überlegen Sie, welche der beiden Formeln für die **Standardabweichung besser geeignet** ist, um mittels eines Programms realisiert zu werden? – **Begründen** Sie Ihre Antwort!

- Zum **Einlesen** eines **Zeichens** für die **Menü-Auswahl** und für die **Steuerung der Textdatei-Ausgabe** kann (muss aber nicht) der folgende **parametrisierte Makro** sinnvoll **eingesetzt** werden:

```
#define getSelection(c)    while((c=getchar())=='\n')
```

  Erläutern Sie die **Funktionalität** dieses Makros! (auch , wenn Sie ihn nicht einsetzen wollen)

- Alle im Programm verwendeten **Konstanten** sind **symbolisch** zu definieren.

- Innerhalb des Programms dürfen **keine Arrays** zur Zwischenspeicherung der Messdaten und/oder der Berechnungsergebnisse eingesetzt werden. **Globale Variable** dürfen ebenfalls **nicht** verwendet werden.

Aufgaben :

a) Erstellen Sie eine **C-Headerdatei `messdatfunc.h`**, die die Definition des obigen Structure-Typs, die verwendeten Konstantendefinitionen, die Definition des obigen Makros sowie die Deklarationen der Funktion zum Überprüfen der Datei-Existenz und der Menü-Funktionen enthält.

b) Beschreiben Sie den von der Funktion zum **Menüpunkt (2)** zu realisierenden Algorithmus durch ein **Struktogramm**!

c) Implementieren Sie die drei **Menüfunktionen** und die **Funktion** zur **Überprüfung auf Datei-Existenz** in der **C-Quelldatei `messdatfunc.c`** !
  Falls sinnvoll können **Teilfunktionalitäten** in **weitere Funktionen** ausgelagert werden, die aber **außerhalb** des **Moduls** nicht verwendbar sein sollen.

d) Erstellen Sie eine **weitere** von den Menüfunktionen getrennte **C-Quelldatei**, in der die **`main()`–**Funktion des Programms implementiert ist.

e) Erstellen Sie das **ablauffähige Programm** aus den beiden o.a. Quelldateien und der Headerdatei!