Übungsaufgaben A01 – A03

A01 – Wiederholung zur Programmierung in C

Schreiben Sie verschiedene Funktionen, die eine Wertereihe prüfen! Die Wertereihe wird als double-Feld (feld) und die Anzahl der Elemente als int-Wert (n) übergeben. Testen Sie die Funktionen in einem Hauptprogramm mit entsprechenden vorinitialisierten Feldern, die Ihnen unten bereitgestellt werden!

Eine der hier beschriebenen Funktionen war nach zufälliger Auswahl Gegenstand der Online-Klausur.

a) eine Funktion `alle_gleich()`, die prüft, ob alle Elemente im Feld wertmäßig gleich sind. Gleichheit wird angenommen, wenn kein Wert von einem anderen um mehr als 0.001 abweicht. Die erfüllte Eigenschaft soll mit der Rückgabe von 1 signalisiert werden. Anderenfalls ist 0 zurückzugeben. Ist die Anzahl der Elemente kleiner als 2, so ist 0 zurückzugeben.

b) eine Funktion `pruefe_sort()`, die prüft, ob alle Elemente mit aufsteigender Indizierung wertmäßig aufsteigend sind. Die erfüllte Eigenschaft soll mit der Rückgabe von 1 signalisiert werden. Anderenfalls ist 0 zurückzugeben. Ist die Anzahl der Elemente kleiner als 2, so ist 0 zurückzugeben.

c) eine Funktion `max_delta()`, welche die maximale absolute Differenz zwischen im Feld vorkommenden Elementen bestimmt und zurückgibt. Wenn weniger als zwei Elemente im Feld übergeben werden, dann ist 0.0 zurückzugeben.

d) eine Funktion `alle_in_intervall()`, die prüft, ob alle Werte größer als eine untere Grenze und kleiner als eine Obergrenze sind. Die erfüllte Eigenschaft soll mit der Rückgabe von 1 signalisiert werden. Anderenfalls ist 0 zurückzugeben. Wird kein Element übergeben (n=0), so ist -1 zurückzugeben.

e) eine Funktion `alle_vz_gleich()`, die prüft, ob alle Elemente das gleiche Vorzeichen aufweisen. Der Wert 0 wird für diese Entscheidung als positiv gewertet. Die erfüllte Eigenschaft soll mit der Rückgabe von 1 signalisiert werden. Anderenfalls ist 0 zurückzugeben. Ist die Anzahl der Elemente kleiner als 2, so ist 0 zurückzugeben.

Funktionsschnittstellen:

```
int alle_gleich(double *feld, int n);
int pruefe_sort(double *feld, int n);
double max_delta(double *feld, int n);
int alle_in_intervall(double *feld, int n, double u_grenze, double o_grenze);
int alle_vz_gleich(double *feld, int n);
```

Vorschlag für Felddeklarationen und Initialisierungen:

```
int n=4;
double werte_praktisch_gleich[] ={ 9.9995, 10.0, 10.0002, 9.99991};
double werte_sort_0123[] = {0, 1, 2, 3};
double werte_unsortiert[] = {1, 3, 2, 0};
double werte_maxabw_0komma1[] = { 1.0, 1.05, 0.95, 1.01};
double werte_gleiche_vz[] = {0, 1, 2, 3};
double werte_verschiedene_vz[] = {-1, 0, 1, 2};
```

A02 – Modularisierung von Programmen

Fassen Sie die in der Aufgabe A01 geschriebenen Funktionen in einem Modul zusammen, das aus einer C-Quelltextdatei und einer Headerdatei besteht! Dafür werden die Bezeichnungen `feldfunktionen.c` und `feldfunktionen.h` vorgeschlagen.

Binden Sie die Dateien in geeigneter Weise in Ihr Projekt ein, und rufen Sie die Funktionen aus einer Hauptquelltextdatei heraus auf!

A03- Dynamische Speicherallokation

Erstellen Sie ein C-Programm, das

- eine Anzahl n als ganzzahligen, positiven Wert von der Konsole als Eingabe entgegennimmt.
- danach einen Speicherbereich für n double-Werte allokiert.
- schrittweise n double-Werte von der Konsole einliest und auf dem allokierten Speicherbereich nacheinander ablegt.
- Die Werte in entgegengesetzter Reihenfolge auf der Konsole ausgibt
- Den Speicherbereich für die n double-Werte wieder freigibt

Die Schnittstellen der nutzenden Standardbibliotheksfunktionen sind:

```
#include <malloc.h>

void *malloc(size_t size);      // gibt die Anfangsadresse des
                                // allokierten Bereichs zurück
void free(void *addresse);      // Freigabe des Speicherbereichs
                                // mit der übergebenen
                                // Anfangsadresse
```