Ingenieurinformatik

C-Programmierung

Name	Vorname	Semestergruppe	Hörsaal

	Aufgabe 1	Aufgabe 2	Aufgabe 3	Aufgabe 4	Aufgabe 5	Summe

Bachelorstudiengang:

- ❒ **Studienbeginn vor WS13/14 (Kombinationsprüfung) ****
- ❒ **Studienbeginn ab WS13/14 bis WS15/16 ****
- ❒ **Studienbeginn ab SS16 (Kombinationsprüfung)**

- ❒ **Diplomstudiengang Fahrzeugtechnik ****

**** Die Prüfung ist nur dann gültig, wenn Sie die Zulassungsvoraussetzung erworben haben (erfolgreiche Teilnahme am Praktikum).**

Aufgabensteller: **Dr. Reichl, Dr. Küpper und Kollegen**

Bearbeitungszeit: **60 Minuten**

Hilfsmittel: **Taschenrechner nicht zugelassen,**
PC/Notebook nicht zugelassen,
sonstige eigene Hilfsmittel sind erlaubt,
Bearbeitung mit Bleistift ist erlaubt.

****** Viel Erfolg! ******

Aufgabe 1: (ca. 22 Punkte)

Erstellen Sie ein C-Programm, welches für n Messwertpaare x_i, y_i (mit $i = 1 \dots n$) die dazugehörige Ausgleichsgerade $y = m \cdot x + b$ berechnet.

- Nach dem Programmstart wird der Anwender nach der Anzahl n der Messwertpaare gefragt. Die Anzahl muss zwischen 3 und 100 liegen, ansonsten wird die Eingabe wiederholt. Es darf aber davon ausgegangen werden, dass der Anwender nur Zahlen (im korrekten Format) eingibt.
- Nun wird der Anwender der Reihe nach zur Eingabe der einzelnen x- und y-Werte aufgefordert (Datentyp **double** verwenden; siehe Bildschirmfoto unten auf dieser Seite!).
- Anschließend werden der Mittelwert der x-Werte (im Folgenden als $\bar{x}$ bezeichnet) und der Mittelwert der y-Werte (im Folgenden als $\bar{y}$ bezeichnet) mit vier Nachkommastellen ausgegeben.
- Schließlich berechnet das Programm die Koeffizienten m und b der Ausgleichsgeraden und gibt beide Werte ebenfalls mit vier Nachkommastellen aus.
- Alle Eingabeaufforderungen und Ausgaben sollen so realisiert werden, wie es im Bildschirmfoto unten auf dieser Seite gezeigt ist.

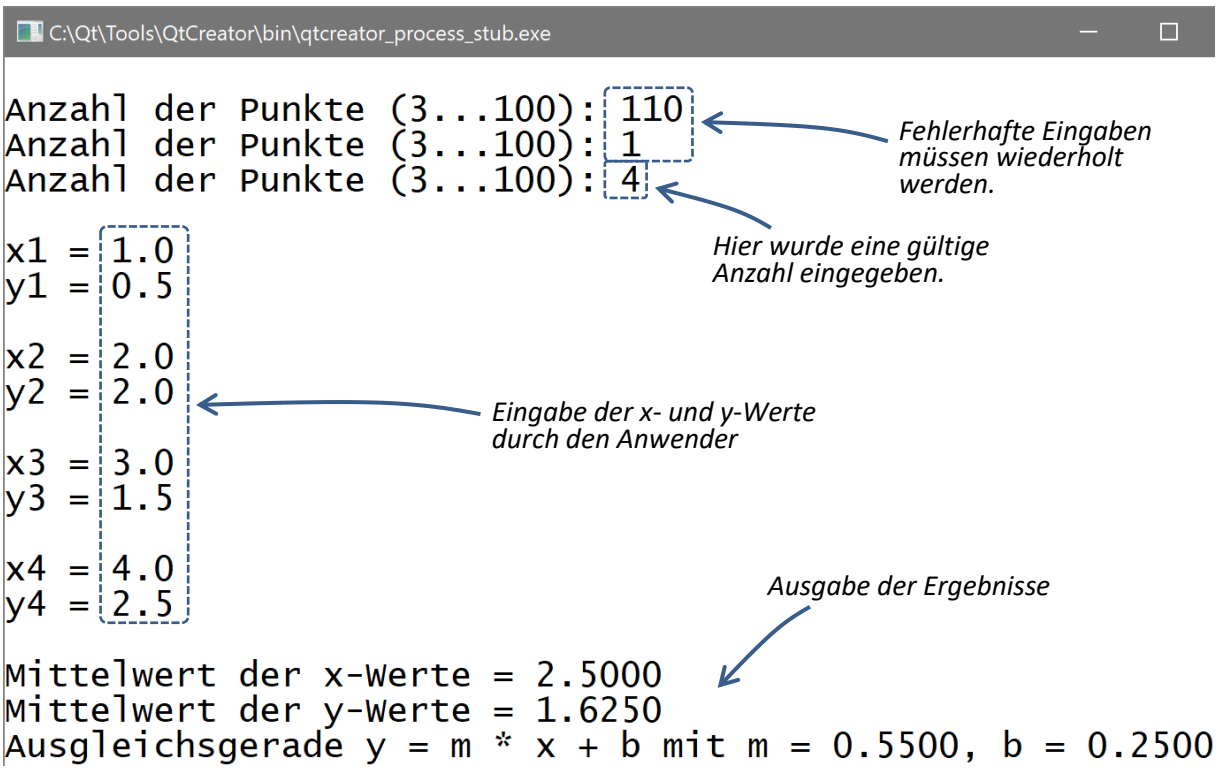
Für den Koeffizienten m gilt:

$$m = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Für den Koeffizienten b gilt:

$$b = \bar{y} - m \cdot \bar{x}$$

Die Bildschirmausgabe des C-Programms soll folgendermaßen aussehen:



Aufgabe 2: (ca. 22 Punkte)

Die globale Variable **data** ist wie folgt definiert: **double data[4][8][32];**

2.1. Schreiben Sie eine C-Funktion **fill1**, welche alle Elemente von **data** auf den Wert 1 (eins) setzt. Schreiben Sie Ihre Funktion unter Verwendung geeigneter **for-Schleifen**.

2.2. Erstellen Sie ein Struktogramm, welches den Ablauf der Funktion **fill1** zeigt.

Lösung zu 2.1 (Quelltext)

```
void fill1(void)
{




}
```

Lösung zu 2.2 (Struktogramm)

2.3. Schreiben Sie nun eine C-Funktion **fill2**, welche alle Elemente von **data** auf den Wert 2 (zwei) setzt. Schreiben Sie diese Funktion unter Verwendung geeigneter **do-while-Schleifen**.

2.4. Erstellen Sie auch ein Struktogramm, welches den Ablauf der Funktion **fill2** zeigt.

Lösung zu 2.3 (Quelltext)

```
void fill2(void)
{




}
```

Lösung zu 2.4 (Struktogramm)

Aufgabe 3: (ca. 10 Punkte)

Schreiben Sie eine Funktion **woerter_zaehlen**, welche die Anzahl der Wörter in einer Zeichenkette ermittelt und als Rückgabewert zurückgibt. Die einzelnen Wörter bestehen aus mindestens einem Buchstaben und sind durch ein (oder mehrere!) Leerzeichen voneinander getrennt.

Ihre Funktion soll auch dann ein korrektes Ergebnis liefern, wenn sich am Anfang der Zeichenkette Leerzeichen befinden oder wenn die Zeichenkette leer ist.

Hinweis: Zur Vereinfachung dürfen Sie davon ausgehen, dass sich in der Zeichenkette lediglich Buchstaben und Leerzeichen befinden – keine anderen Zeichen.

```
#include <stdio.h>
int woerter_zaehlen(char *s);

int main(void)
{
    printf("%d\n", woerter_zaehlen("a bb ccc"));    // Ausgabe: 3
    printf("%d\n", woerter_zaehlen("     "));       // Ausgabe: 0
    printf("%d\n", woerter_zaehlen(""));            // Ausgabe: 0
    printf("%d\n", woerter_zaehlen("  xx  yy  "));  // Ausgabe: 2
    return 0;
}

int woerter_zaehlen(char *s)
{









}
```

Aufgabe 4: (ca. 7 Punkte)

4.1. Das folgende C-Programm berechnet die Bewegung eines Feder-Masse-Schwingers mit dem Eulerverfahren. (Das zu lösende Differentialgleichungssystem besteht aus zwei Differentialgleichungen erster Ordnung.) Korrigieren Sie die fünf Fehler im C-Quelltext!

```
/* Schwingungs-DGL mit Eulerverfahren lösen */
#include <stdio.h>

#define DELTA_T 0.001;  /* Zeitschritt in Sekunden */
#define SCHRITTE 10000  /* Anzahl der Zeitschritte */

void main(void)
{
    double y[1] = { 1.0, 0.0 };  /* Anfangswerte */
    double k = 1.0, m = 1.0;     /* Federkonstante, Masse */
    double y_punkt[2], t = 0;

    int i;
    for(i = 0; i < SCHRITTE; ++i)
    {
        /* Ableitung berechnen */
        y_punkt[0] = y[1];
        y_punkt[1] = -k/m * y[0];

        /* Eulerverfahren */
        y[0] = y[0] + DELTA_T * y_punkt[0];
        y[1] = y[1] + DELTA_T * y_punkt[2];
        t = t + DELTA_T;

        /* Aktuelle Zeit und Auslenkung ausgeben */
        printf("%.4f ; %.4f \n, t, y[0]");
    }

    /* XXX Welchen Wert hat i an dieser Stelle? XXX */
    return 0;
}
```

4.2. Welchen Wert hat i nach dem Ende der for-Schleife, also in der mit /* XXX */ markierten Zeile?

i = ____

4.3. Wie oft werden die Befehle innerhalb der for-Schleife durchlaufen?

Anzahl der Schleifen-Durchläufe = ____

Aufgabe 5: (ca. 6 Punkte)

5.1. Wandeln Sie die folgende Dezimalzahl in eine Dualzahl (Binärsystem) um. Geben Sie nicht nur das Endergebnis, sondern auch alle Zwischenschritte Ihrer Berechnung an!

199_{DEZ} = ??$_{BIN}$

5.2. Wandeln Sie dieselbe Zahl 199_{DEZ} ins Hexadezimalsystem um!

199_{DEZ} = ??$_{HEX}$

(Platz für Notizen und Nebenrechnungen)