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. 29 Punkte)

Erstellen Sie ein C-Programm zur Auswertung von bis zu 100 gültigen Messwerten:

- Nach dem Programmstart werden die Messwerte der Reihe nach eingegeben.
- Die Messwerte sollen als double-Werte abgespeichert und verarbeitet werden.
- Messwerte, die größer als 100 sind, werden ignoriert. Es handelt sich dabei um ungültige Fehlmessungen, die nicht abgespeichert und weiter verarbeitet werden.
- Nachdem 100 gültige Messwerte eingegeben wurden – oder – wenn ein negativer Wert eingegeben wurde, endet die Eingabe.
- Die Liste der gültigen Messwerte wird zur Kontrolle auf dem Bildschirm ausgegeben.
- Auch der Mittelwert $\bar{x}$ der gültigen Messwerte wird auf dem Bildschirm ausgegeben.
- Anschließend wird die Standardabweichung σ der gültigen Messwerte berechnet und ausgegeben.
- Die Ausgabe der Messwerte, des Mittelwerts und der Standardabweichung σ soll mit 3 Nachkommastellen und einer Feldbreite von 8 Zeichen erfolgen.
- Falls gar keine gültigen Messwerte eingegeben werden, soll eine entsprechende Meldung ausgegeben und das Programm beendet werden.

Formel zur Berechnung der Standardabweichung (n = Anzahl, $\bar{x}$ = Mittelwert der Messwerte):

$$\text{Standardabweichung } \sigma = \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - \bar{x})^2}$$

Die Bildschirmausgabe des Programms soll folgendermaßen aussehen:

```
C:\WINDOWS\system32\cmd.exe
1. Messwert: 6
2. Messwert: 2
3. Messwert: 150
Fehlmessung wird ignoriert!
3. Messwert: 10
4. Messwert: 24
5. Messwert: 200
Fehlmessung wird ignoriert!
5. Messwert: 18
6. Messwert: -5

Liste der Messwerte:
1. Messwert:    6.000
2. Messwert:    2.000
3. Messwert:   10.000
4. Messwert:   24.000
5. Messwert:   18.000

Mittelwert  :   12.000
Standardabw.:    8.000
```

Eingabe der Messwerte

Ausgabe der gültigen Messwerte, Ausgabe von Mittelwert und Standardabweichung

```
C:\WINDOWS\system32\cmd.exe
1. Messwert: 150
Fehlmessung wird ignoriert!
1. Messwert: -10

Keine gueltigen Messwerte!
```

Keine gültigen Messwerte, Meldung wird ausgegeben

Aufgabe 2: (ca. 10 Punkte)

Suchen Sie die Fehler in den folgenden Quelltexten!

2.1. Die Funktion count_space() zählt die Leerzeichen in der Zeichenkette, die als Parameter übergeben wird und gibt das Ergebnis als Rückgabewert zurück: **(3 Fehler)**

```
int count_space(char *str)
{
    int i = 0, count;
    while(str[i] != '0')
    {
        if(str[i] == ' ')
            ++count;
    }
    return count;
}
```

2.2. Das folgende Programm berechnet das Skalarprodukt von a[] und b[] und gibt das Ergebnis auf dem Bildschirm aus: **(3 Fehler)**

```
#include <stdio.h>
#define N = 3

int main(void)
{
    int i;
    double s = 0;
    double a[N] = { 1, 3, 9};
    double b[N] = { 2, 2, 2};

    for(i = 1; i < N; ++i)
        s += a[i] * b[i];

    printf("Skalarprodukt: %d\n", s);
    return 0;
}
```

2.3. Die Funktion punkt_im_einheitskreis() prüft, ob sich der Punkt P(x,y) innerhalb des Einheitskreises befindet (das ist ein Kreis mit Radius = 1 um den Koordinaten-Ursprung). Falls ja, ist der Rückgabewert = 1, ansonsten ist der Rückgabewert = 0. Die Funktion gibt außerdem den Abstand des Punkts P(x,y) vom Ursprung mittels Zeiger zurück: **(3 Fehler)**

```
int punkt_im_einheitskreis(double x, double y, double abstand);
{
    *abstand = sqr(pow(x, 2) + pow(y, 2));
    if(*abstand < 1.0)
        return 1;
    else
        return 0;
}
```

2.4. Wie sieht der Aufruf der (korrigierten) Funktion punkt_im_einheitskreis() aus, wenn der Rückgabewert in der Variablen ok und der Abstand in der Variablen a gespeichert werden soll?

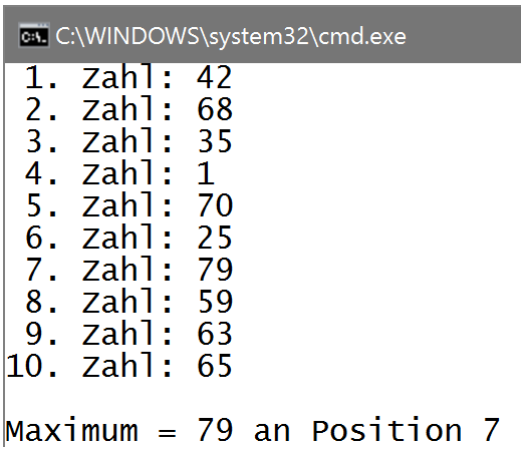
```
int ok;
double a, x = 0.46, y = 0.88;
```

ok =

Aufgabe 3: (ca. 10 Punkte)

Zeichnen Sie ein Struktogramm (!!) eines C-Programms, das die folgenden Aufgaben erfüllt:

- Es werden 10 ganzzahlige Zufallszahlen im Bereich von 1…100 erzeugt und ausgegeben.
- Das Programm ermittelt die größte (Maximum) der Zufallszahlen und auch die Position in der Zahlenfolge, wo sich das Maximum befindet.
- Position und Maximum werden ebenfalls auf dem Bildschirm ausgegeben.
- Die Bildschirmausgabe des Programms soll wie in der nebenstehenden Abbildung aussehen.



Tipps: Die Zahlen sollen nicht gespeichert werden!
Lösen Sie die Aufgabe ganz einfach ohne Verwendung von Feldern/Vektoren/Arrays!

Struktogramm:

Aufgabe 4: (ca. 8 Punkte)

Eine globale Matrix m ist wie folgt definiert:

```
#define N 10
double m[N][N];
```

Schreiben Sie eine Funktion diagonal(), welche die Matrix m wie folgt belegt: Alle Elemente auf der Hauptdiagonalen und alle Elemente auf den beiden Nebendiagonalen sollen mit unterschiedlichen Zufallszahlen (Datentyp: double) im Bereich von -0.5…+0.5 belegt werden.

```
void diagonal(void)
{




}
```

Aufgabe 5: (ca. 10 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!

$200_{DEZ} = ??_{BIN}$

5.2. Wandeln Sie dieselbe Zahl 200_{DEZ} ins Hexadezimalsystem um!

$200_{DEZ} = ??_{HEX}$

(Platz für Notizen und Nebenrechnungen)