Ingenieurinformatik

Name	Vorname	Matrikelnummer	Sem.-Gr.:	Hörsaal	Platz

Zulassung geprüft
vom Aufgabensteller:

_____

Teil 1	Aufg. 2	Aufg. 3	Aufg. 4	Summe	Note

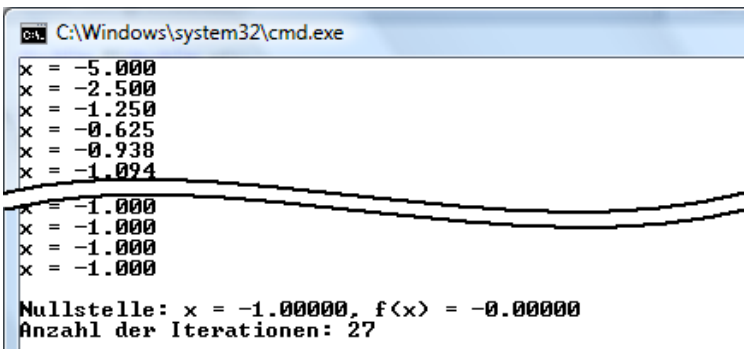
Teil 1/Aufgabe 1: 30 Minuten ohne Unterlagen, **Teil 2/Aufgaben 2-4:** 60 Minuten, beliebige eigene Unterlagen aber keine PC/Laptops, Bearbeitung mit Bleistift erlaubt.

Die Prüfung ist nur dann gültig, wenn Sie die erforderliche Zulassungsvoraussetzung erworben haben (drei Testate im Praktikum). Dies wird vom Aufgabensteller überprüft.

Aufgabe 2: (ca. 22 Punkte)

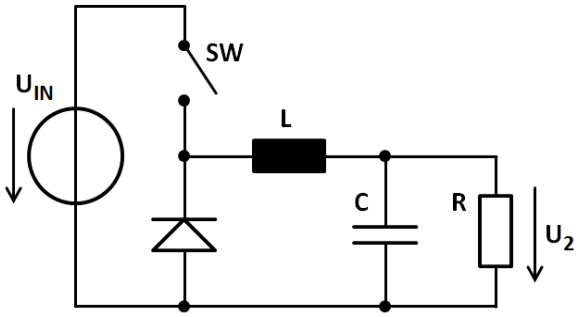
Die Funktion $f(x) = 3x^3 - 5x^2 + 8$ besitzt im Intervall [-10…0] genau eine Nullstelle. Erstellen Sie ein C-Programm zur numerischen Bestimmung dieser Nullstelle mit dem Bisektionsverfahren. Sie benötigen dazu drei Variablen x, x1 und x2:

1. Das zu durchsuchende Intervall [-10…0] wird den Variablen x1, x2 zugewiesen: x1 = -10, x2 = 0.
2. Bestimmen Sie einen Näherungswert x = ½ (x1 + x2) für die gesuchte Nullstelle und geben Sie diesen Näherungswert mit drei Nachkommastellen auf dem Bildschirm aus.
3. Falls der Betrag $|f(x)| < 10^{-6}$ ist, geben Sie x und f(x) mit fünf Nachkommastellen auf dem Bildschirm aus! Die gesuchte Nullstelle wurde mit ausreichender Genauigkeit bestimmt, das Bisektionsverfahren wird abgebrochen. Ansonsten wird das Verfahren mit Unterpunkt 4 fortgesetzt.
4. Falls das Produkt $f(x1) \cdot f(x) < 0$ ist, muss die gesuchte Nullstelle im Intervall [x1…x] liegen. Führen Sie in diesem Fall die Zuweisung x2 = x durch. Ansonsten führen Sie die Zuweisung x1 = x durch. Das Verfahren wird in beiden Fällen mit Unterpunkt 2 fortgesetzt.
5. Bevor das Programm beendet wird, erfolgt auf dem Bildschirm eine Ausgabe, wie viele Iterationen notwendig waren, um die Nullstelle zu ermitteln. (Die Berechnung des ersten Näherungswerts wird als erste Iteration gezählt.)



Aufgabe 3: (ca. 23 Punkte)

Die Abbildung zeigt die Schaltung eines DC-DC-Wandlers. Der Schalter SW wird mit einer Frequenz von 10 kHz und einem Tastgrad von 50% betätigt.



Zur Simulation des Spannungsverlaufs von U_2 am Verbraucher R wurde das folgende C-Programm erstellt.

```
#include "chart.h"
#include <stdio.h>

#define L 0.0024
#define C 0.00001
#define R 10.0
#define U_IN 24.0
#define T_DELTA 0.0000001
#define T_END 0.005

#define FREQ = 10000.0

int main(void);
{
    int sw, winkel;
    double t, u2, y;

    for(t = 0, u2 = 0, y = 0; t <= T_END; t += T_DELTA)
    {
        chart_lineto(t, u2, CHART_BLUE);

        sw = 0;
        winkel = (int)(360 * t * FREQ) % 360;
        if(winkel > 180) sw = 1;

        switch(sw);
        {
        case 0:  y += T_DELTA * (U_IN / L - u2 / L - y / R) / C;
                 u2 += T_DELTA * y;
                 break;
        case 1:  y += T_DELTA * (-u2 / L - y / R) / C;
                 u2 += T_DELTA * y;
                 break;
        default  printf("Interner Fehler!\n");
                 return 0;
        }
    }
    chart_show(CHART_DEFAULT, "DC-DC-Wandler, Ausgangsspannung");
}

return 0;
```

siehe Aufgabe 3.3

3.1. Korrigieren Sie die fünf Fehler, die sich in den Quelltext eingeschlichen haben.

3.2. Zeichnen Sie für das (korrigierte) Hauptprogramm ein Struktogramm, welches den genauen Ablauf der Funktion **main** beschreibt.

Struktogramm:

Symbolische Konstanten:	L = 0,0024; C = 0,00001; R = 10,0; U_IN = 24,0; T_DELTA = 0,0000001 T_END = 0,005; FREQ = 10000,0
int-Variablen: double-Variablen:	sw, winkel t, u2, y

Fortsetzung von Aufgabe 3: **Name:** _____

3.3. Im C-Programm wurde die switch-case-Anweisung verwendet, um je nach Wert der Variablen sw drei unterschiedliche Aktionen durchzuführen. Programmieren Sie diese Verzweigungen auf eine andere Art ohne Verwendung von switch-case! (Hinweis: Nur die neuen Verzweigungen hinschreiben, kein komplettes Programm.)

Aufgabe 4: (ca. 22 Punkte)

4.1. Wie lautet die Ausgabe des folgenden Programms:

```c
#include <stdio.h>
#include <string.h>
#include <ctype.h>

void count(char *text, int *g, int *k);

int main(void)
{
    int g, k;
    char text[] = "abc DEF ghi 123";
    count(text, &g, &k);
    printf("%s\n%d, %d", text, g, k);
    return 0;
}

void count(char *text, int *g, int *k)
{
    int i;
    *g = 0; *k = 0;
    for(i = 0; i < strlen(text); ++i)
    {
        if(islower(text[i])) *k += 1;
        if(isupper(text[i])) *g += 1;
    }
}
```

Ausgabe:

4.2. Eine globale Matrix ist wie folgt definiert: **double m[10][10];**

Schreiben Sie eine Funktion **void hilbert(void)**, welche alle Elemente dieser Matrix so belegt, dass sich eine Hilbertmatrix zehnter Ordnung ergibt:

$$H_{10} = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{10} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{11} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots & \frac{1}{12} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \cdots & \frac{1}{19} \end{pmatrix}$$

```
double m[10][10];
void hilbert(void)
{



}
```

4.3. Die Funktion **void vec(int anz, double *v, double *max, double *sum)** bekommt einen Vektor **v** mit **anz** Elementen übergeben. Sie ermittelt das größte Element des Vektors und berechnet auch die Summe aller Elemente. Beide Ergebnisse werden über die Parameter **max** bzw. **sum** zurückgegeben. Schreiben Sie den C-Quelltext dieser Funktion! (Hinweis: Sie dürfen ohne weitere Überprüfung davon ausgehen, dass **anz > 0** ist.)

```
void vec(int anz, double *v, double *max, double *sum)
{



}
```