Matlab Tutorials

Das *LiveScript* zu dem Tutorial 6.

Lösung: Aufgabe aus Tutorial 5

```
x = linspace(0, 8*pi, 100);
y = cos(x) + 1i * sin(x);
plot3(x, real(y), imag(y))
ha = gca;

ha.XGrid = 'on';
ha.XTick = 0:pi:8*pi;
ha.XTickLabel = {'0 pi', '1 pi', '2 pi', '3 pi', '4 pi', '5 pi', '6 pi', '7 pi', '8 pi'};
ha.XLim = [min(x) max(x)];
ha.YLabel.String = 'Re\{y\}';
ha.ZLabel.String = 'Im\{y\}';
set(findall(ha, '-property', 'FontSize'), 'FontSize', 14)

az = linspace(-37.5, 0, 50);
el = linspace(30, 0, 50);
for n = 1:50
    view([az(n) el(n)])
    drawnow
    pause(0.1)
end
```

Tutorial 6.1

- Einführung in Klassen

```
% sichtbare Eigenschaften
properties (Access = public)
    wert1
end

% unsichtbare Eigenschaften
properties (Access = protected)
    wert2 = 5;
end
```

```
%% Methoden
methods
    % Konstruktor
    function self = BasicClass(val)
        self.wert1 = val;
    end

    % weitere Methoden
    function r = abrunden(self)
        r = floor(self.wert1);
```

```
    end
end
```

Tutorial 6.2

- Beispiel AUFile

Mit AUFile werden wir ein paar Anwendungsfälle von OOP in Matlab durchgehen.

```
% Erzeugung eines Objektes
obj = AUFile('test.au', 'new');

% Übersicht des Objektes
disp(obj)

% Eigenschaften betrachten
properties(obj)

% Methoden betrachten
methods(obj)
```

Eigenschaften verwenden:

```
dur_sec = 5;
data    = rand(obj.SampleRate * dur_sec, 2) - 0.5;
```

write-Methode verwenden um Daten in die Datei zu schreiben:

```
obj.write(data)
```

readall-Methode, um die gesamte Datei einzulesen:

```
obj.readall
```

Mit der AUFile Klasse ist es möglich Dateien blockweise zu lesen ohne dass die Datei für jeden Lesevorgang neu geöffnet/geschlossen wird.

```
for n = 1:3
    data_tmp = obj.read(5);
    disp(data_tmp)
end
```

Tutorial 6.3

- Besprechung von Code in AUFile.m

Tutorial 6.4

- Kleines akustisches Beispiel

Schreiben

```
clear
vFreq = [440, 441];     % Beide Frequenzen für Schwebung
dur_sec= 3;             % Dauer des Signals
fs = 44100;             % Sampling Frequenz
vTime  = linspace(0, dur_sec, fs*dur_sec); % vector containing the time, starting at t=0

sinus = zeros(length(vTime), 2);
for n = 1:2
    sinus(:, n) = sin(2 * pi * vFreq(n) * vTime);
end

signal = [sinus(:,1) + sinus(:,2)];  % Mono: Überlagerung der Sinus Signale
signal = signal / max(abs(signal)) * 0.9; % Normierung (auf 0.9) => kein Clipping
plot(vTime, signal)                  % grafische Darstellung der Schwebung

obj = AUFile('schwebung.au', 'new', size(signal, 2), fs);
obj.write(signal)

clear
```

Einlesen

```
obj = AUFile('schwebung.au', 'read');

% ACHTUNG: Bevor man sich Audiosignale ausgeben lässt, IMMER die Lautstärke herunterdrehen
% und schrittweise erhöhen!
% soundsc(obj.readall, obj.SampleRate);
% soundsc(obj.readall, obj.SampleRate / 2);
% soundsc(obj.readall, obj.SampleRate * 2);
```

Tutorial 6: Aufgabe

Mit Hilfe der anonymen Funktionen in Matlab drei Gleichungen entwickeln, die den Strom, die Spannung und den Widerstand nach dem Ohmschen Gesetz berechnen (also: U=R*I; R=...; I=...).

```
U = @( ...
R = @( ...
I = @( ...
```