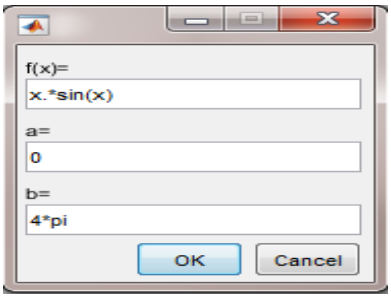
Dr. Frank Schöpfer

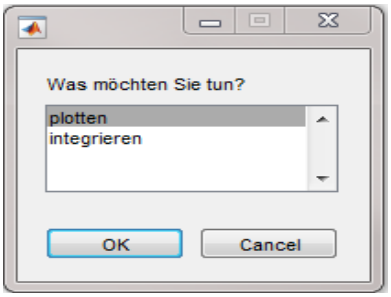
Einführung in Matlab
Übung 6

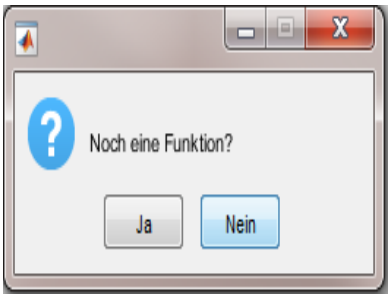
Aufgabe 1: Schreiben Sie ein Skript `funktionseingabe` mit folgenden Eigenschaften:

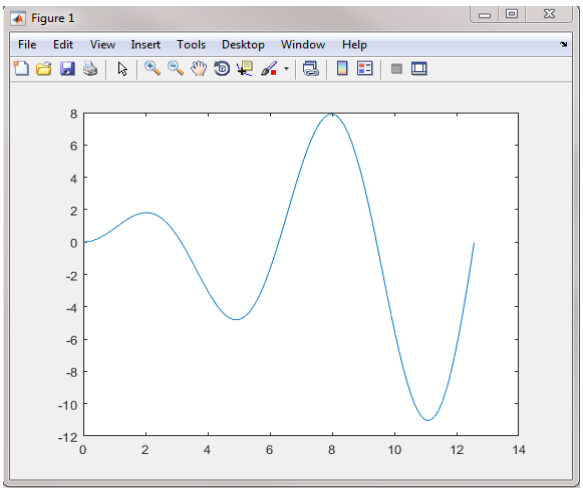
- In einer Eingabebox (`inputdlg`) kann man eine Funktion $f(x)$ und Intervallgrenzen a, b eingeben.
- Danach kann man auswählen (`listdlg`), ob man die Funktion "plotten" oder "integrieren" möchte.
- Je nach Auswahl wird dann das entsprechende getan, wobei bei "integrieren" ein Flächenplot (`area`) erzeugt wird, und im Titel der berechnete Wert des Integrals $\int_a^b f(x)\,dx$ angegeben wird.
- Schließlich wird man gefragt (`questdlg`), ob man noch eine Funktion behandeln möchte.
- Das ganze wird solange wiederholt bis am Ende "Nein" gewählt wird.

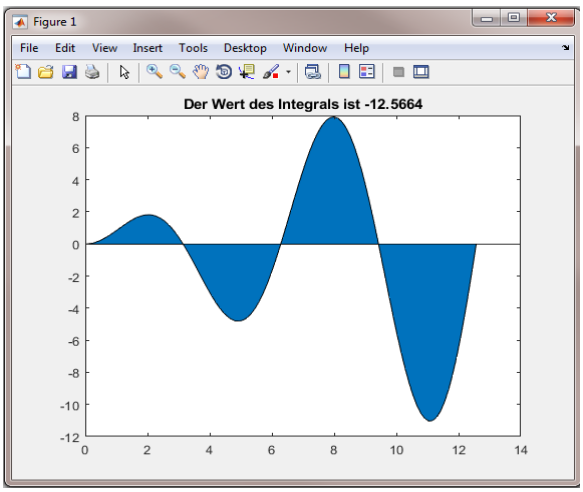
Zur Erinnerung: Für die Umwandlung eines Strings in ein Function Handle dient der Befehl `str2func`.











Aufgabe 2: (Übung zum Unterschied zwischen eckigen, runden und geschweiften Klammern.
Achtung: Dies ist vermutlich die schwierigste Aufgabe dieser Übung.)
Schreiben Sie eine rekursive Funktion `P=potenzmenge(C)`, die von einer Menge C die Potenzmenge $P(C)$ von C bestimmt (Menge aller Teilmengen von C; hier ohne leere Menge). Dabei sollen C und P Cell Arrays sein, und P wie folgt rekursiv konstruiert werden:

- Hat C nur ein Element, so ist $P = \{C\}$.
- Sonst berechne Potenzmenge Q von $C \setminus \{c\}$ mit einem $c \in C$ (d.h. Potenzmenge der Menge mit einem Element weniger),
- und setze dann P geeignet zusammen aus: Q, $\{c\}$, und allen Mengen in Q jeweils vereinigt mit c

Test

```
>> P=potenzmenge({'Birne','Apfel','Banane'})
P =
  1x7 cell array
    {1x3 cell} {1x2 cell} {1x2 cell} {1x1 cell} {1x2 cell} {1x1 cell} {1x1 cell}
>> P{:}
ans =
  1x3 cell array
    {'Birne'}    {'Apfel'}    {'Banane'}
ans =
  1x2 cell array
    {'Apfel'}    {'Banane'}
ans =
  1x2 cell array
    {'Birne'}    {'Banane'}
ans =
  1x1 cell array
    {'Banane'}
ans =
  1x2 cell array
    {'Birne'}    {'Apfel'}
ans =
  1x1 cell array
    {'Apfel'}
ans =
  1x1 cell array
    {'Birne'}
```

Aufgabe 3: Schreiben Sie eine Funktion `p=erzeuge_polygon(varargin)`, die eine Structure `p` für ein Polygon erzeugt, wie wir es in der Vorlesung gemacht haben. Dabei soll die Funktion folgende Eigenschaften haben:

- Zunächst wird `p` mit Default-Werten erzeugt:
  Ecken=`[]` (leeres Array), Position=`[0;0]`, Winkel = `0`, Farbe= `'r'` (rot).
- Dann kann man (eine beliebige Anzahl von) Ecken graphisch mit `ginput` eingeben.
- Die Funktion soll mit `varargin` so geschrieben werden, dass eine variable Anzahl von Argumenten übergeben werden kann. Somit kann man wahlweise nichts, oder Paare der Form `'Farbe','b'`, `'Winkel',pi/2` oder `'Position',[3;2]` übergeben, welche dann statt den Default-Werten genommen werden.

Test:

```
>> p=erzeuge_polygon
p =
  struct with fields:

       Ecken: [2x5 double]
    Position: [2x1 double]
      Winkel: 0
       Farbe: 'r'
>> male(p)
>> p=erzeuge_polygon('Winkel',pi/2)
p =
  struct with fields:

       Ecken: [2x4 double]
    Position: [2x1 double]
      Winkel: 1.5708
       Farbe: 'r'
>> male(p)
>> p=erzeuge_polygon('Farbe','b','Winkel',pi/2)
p =
  struct with fields:

       Ecken: [2x3 double]
    Position: [2x1 double]
      Winkel: 1.5708
       Farbe: 'b'
>> male(p)
```