

Einführung in die Informatik I

Kapitel I.9: Arbeitsweise von Funktionen

Prof. Dr.-Ing. Marcin Grzegorzek
Juniorprofessur für Mustererkennung im Institut für Bildinformatik
Department Elektrotechnik und Informatik
Fakultät IV der Universität Siegen

05.12.2012

Inhaltsverzeichnis

- I. MATLAB-Einführung
 - 1. Voraussetzungen und Konventionen
 - 2. Variablen und arithmetische Ausdrücke
 - 3. Automatisierung von Berechnungen
 - 4. Logische Ausdrücke
 - 5. Verzweigungen
 - 6. Schleifen
 - 7. Fehlersuche in Programmen
 - 8. Funktionen
 - 9. Arbeitsweise von Funktionen**
 - 10. Vektoren
 - 11. Matrizen
- II. Algorithmen
- III. MATLAB-Fortsetzung
- IV. Wissenschaftliche Werkzeuge

Merkmale von Funktionen

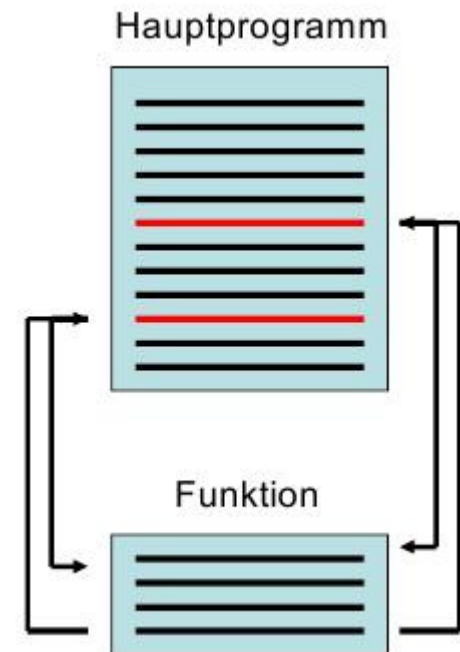
- Funktionen sind vorprogrammierte Programmstücke, die in neue Programme eingebaut werden können.
- Funktionen haben Parameter, mit denen ihr Verhalten von außen beeinflusst werden kann.
- Funktionen können Resultate zurückgeben, müssen aber nicht.

Hauptgründe für Implementierung von Funktionen

- Wiederverwendbarkeit: Dasselbe Codestück kann ohne Mehraufwand immer wieder eingesetzt werden.
- Wartbarkeit: Änderungen in der Funktion wirken sich auf alle ihre Anwendungen gleich aus.
- Übersicht: Teilaufgaben werden als Funktionen ausgelagert, um überschaubare Programmabschnitte zu bekommen.
- Modularität: Verschiedene Programmierer können sich die Arbeit teilen. Jeder arbeitet an **seinem** Programmteil.

Ablauf einer Funktionsausführung

- Sich wiederholende Programmteile können in Funktionen zusammengefasst werden.
- Das Hauptprogramm wird an der Stelle des Funktionsaufrufs unterbrochen und erst fortgesetzt, wenn die Funktion abgearbeitet ist.
- Eine Funktion kann wiederholt von verschiedenen Stellen des Programms aus aufgerufen werden.
- Funktionen haben Parameter, mit denen sich ihr Verhalten von außen steuern lässt.



Beispiel-Funktion zur Illustration

- Eine ganz einfache Funktion mit einem Übergabeparameter und einem Rückgabewert:

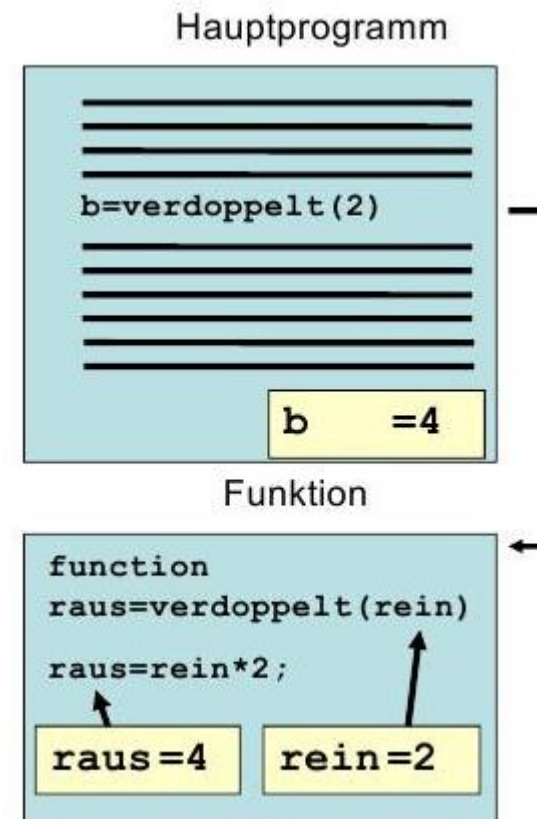
```
function raus = verdoppelt(rein)  
    raus = rein*2 ;
```

- Die Funktion wurde als **verdoppelt.m** gespeichert, dadurch wird der Funktionsumfang von MATLAB um diesen Befehl erweitert.
- Im Commandwindow oder von einem Skript aus kann jetzt die Funktion mit Übergabeparametern gerufen werden:

```
b=verdoppelt (2)           ergibt sich      4  
a=verdoppelt (b)           ergibt sich      8  
a=verdoppelt (a+1)         ergibt sich     18
```

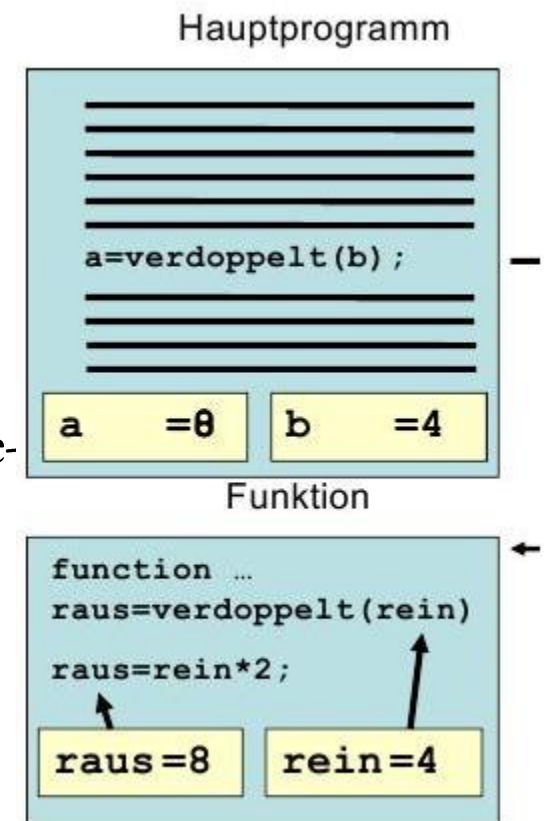
Aufruf mit einer Konstanten

- Beispiel: **b=verdoppelt(2)**
- Die Variablen der Funktion werden im Speicher angelegt.
- Der Wert **2** wird dem Übergabeparameter **rein** zugewiesen.
- Berechnung wird durchgeführt und das Resultat **4** wird der Rückgabevervariablen **raus** zugewiesen.
- Der Quelltext der Funktion ist zu Ende. **verdoppelt(2)** nimmt den Wert der Rückgabevervariablen (**=4**) an und wird **b** zugewiesen.
- Die Variablen der Funktion werden aus dem Speicher entfernt.
- Das Hauptprogramm wird fortgesetzt.



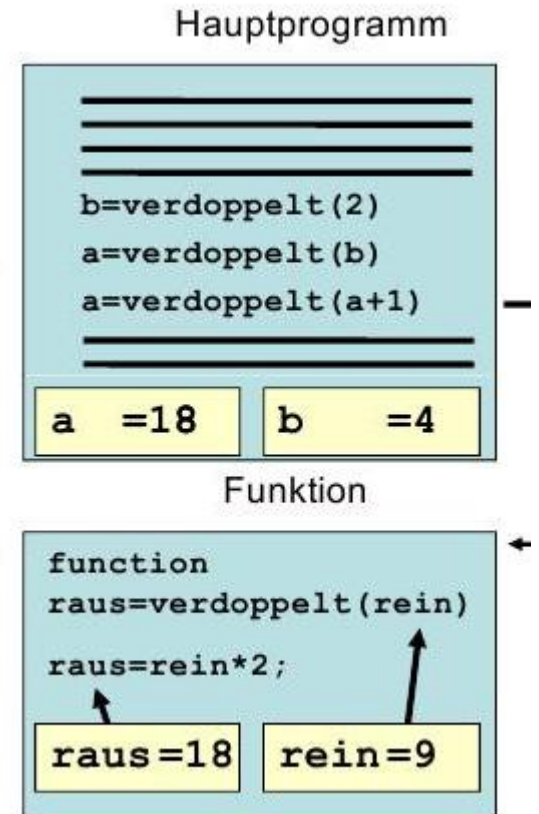
Aufruf mit einer Variablen

- Beispiel: `a=verdoppelt(b)`
- Zwei Variablen `a` , `b` sind aktuell im Speicher.
- Die Variablen der Funktion werden im Speicher angelegt.
- Der Wert von `b (=4)` wird dem Übergabeparameter `rein` zugewiesen.
- Berechnung wird durchgeführt und das Resultat `8` wird der Rückgabewariable `raus` zugewiesen.
- Der Quelltext der Funktion ist zu Ende.
`verdoppelt(b)` nimmt den Wert der Rückgabewariable `(=8)` an und wird `a` zugewiesen.
- Die Variablen der Funktion werden aus dem Speicher entfernt und das Hauptprogramm wird fortgesetzt.



Aufruf mit einem Ausdruck

- Beispiel: `a=verdoppelt(a+1)`
- Die Variablen der Funktion werden im Speicher angelegt.
- Die Berechnung `a+1` wird durchgeführt und das Ergebnis (`=9`) wird dem Übergabeparameter `rein` zugewiesen.
- Berechnung wird durchgeführt und das Resultat `18` wird der Rückgabewariablen `raus` zugewiesen.
- Der Quelltext der Funktion ist zu Ende.
`verdoppelt(a+1)` nimmt den Wert der Rückgabewariablen (`=18`) an und wird `a` zugewiesen.
- Die Variablen der Funktion werden aus dem Speicher entfernt und das Hauptprogramm wird fortgesetzt.

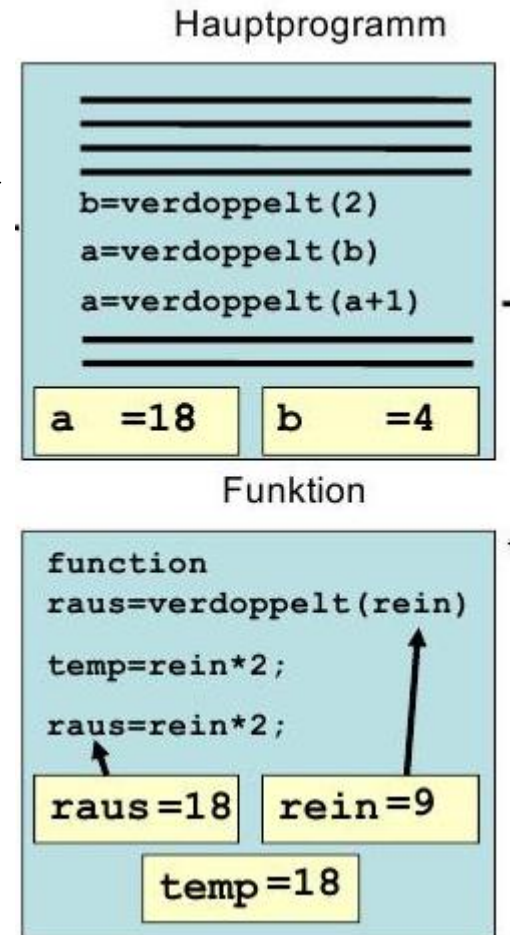


Lokale Variablen

- Lokale Variablen sind Variablen, die innerhalb des Funktionsrumpfs eingeführt werden und sie sind nur innerhalb der Funktion gültig d.h. andere Programmteile können nicht auf diese Variablen zugreifen.
- Wenn die Funktion beendet wird, dann werden auch die lokalen Variablen aus dem Speicher entfernt.

```
function raus=verdoppelt(rein)
    temp = rein*2 ;
    raus = temp ;
```

- Bei erneutem Funktionsaufruf sind daher alte Berechnungsergebnisse in lokalen Variablen **nicht** mehr vorhanden.



warning Befehl

- Falls während der Abarbeitung der Funktion unerwünschte Zustände auftreten, kann dem Benutzer eine **Warnmeldung** im Command Window angezeigt werden.
- Und das Programm wird **NICHT** dadurch **ABGEBROCHEN**.
- Z.B.:

```
function Celsius=F2C(Fahrenheit)
    Celsius=(Fahrenheit-32)/(1.8);
if Celsius < -273.15
        warning('Der absolute Nullpunkt wurde unterschritten. Prüfen Sie Ihre Berechnungen !')
    end
```

```
❖ >> F2C(-480)
```

```
Warning: Der absolute Nullpunkt wurde unterschritten. Prüfen Sie Ihre Berechnungen !
```

error Befehl

- Funktionen können auch durch Fehlermeldung, ABGEBROCHEN werden.
- Der Text der Fehlermeldung erscheint im Command Window (Ohne Striche) [vgl. input- und disp-Befehl]

```
function y=fakultaet(x)
    if x<0
        error ('x muss eine nicht negative ganze Zahl
                sein !') ;
    end
    y=1;
    for i=1:x
        y=y*i
    end
```

```
>> fakultaet(-10)
```

```
??? Error using ==> fakultaet
```

```
X muss eine nicht negative ganze Zahl sein !
```

Informationen

Quelle :

- [1] Vorlesungsskript Einführung in die Informatik I, Prof. Reichhardt, Universität Siegen, 2009.
- [2] A Guide to MATLAB for Beginners and Experienced Users, Brian R. Hunt, Ronald L. Lipsman, Jonathan M. Rosenberg, Cambridge University Press, 2001.
- [3] An Introduction to Programming and Numerical Methods in MATLAB, S.R. Otto and J.P. Denier, Springer, 2005.

Personen:

- Prof. Dr. Marcin Grzegorzek
 - Address: Research Group for Pattern Recognition Department ETI, University of Siegen Hoelderlinstr. 3, H-F 016, D-57076 Siegen.
- Dr. -Ing. Andreas Hoffmann
 - Address: Research Group for Pattern Recognition Department ETI, University of Siegen, Hoelderlinstr. 3, H-B 8405, D-57076 Siegen.