



**WBS
TRAINING**

Programmierung(1)

Agenda

- **(selbsterzeugte) Funktionen(II)** *(Funktionen mit Rückgabewert)*
 - Motivation
 - EVA-Prinzip
 - Vermeidung eines „Seiteneffekts“
 - Beispielaufgabe
 - Darstellung in PAP, Struktogramm, Pseudocode
 - Syntax in ANSI C
- Ausführliches Training + Ergebnisbesprechung
- Fachpraktische Anwendungen

Funktionen(II) – Motivation – EVA-Prinzip

- Wir haben Funktionen auch schon als „(Unter)-**Programme**“ bezeichnet und bei Programmen im allgemeinen über das sogenannte **EVA-Prinzip** gesprochen.
- Dieses Prinzip können wir nun auch vollständig bei Funktionen abbilden:
 - E(ingabe) bei Funktionen geschieht durch die **Übergabewerte**
 - V(erarbeitung) bei Funktionen entspricht der **Funktionalität** des Funktions-Codes
 - A(usgabe) lernen wir heute in Form eines **Rückgabewertes** kennen

Hinweise:

- Für die Anzahl der Übergabewerte gibt es nach oben (zumindest theoretisch) keine Grenzen.
- Pro Funktion kann es aber stets **nur genau einen Rückgabewert** geben.
- Das Zurückgeben eines Rückgabewertes ist gleichbedeutend mit dem Ende der Funktion (und dem weiteren Abarbeiten der Hauptfunktion).
- Der Rückgabewert wird genau an jener Stelle zurückgegeben, von der die Funktion aufgerufen worden war.

Funktionen(II) – Motivation – Seiteneffekt

- Wir hatten bisher ausschließlich Funktionen kennen gelernt, die alleine durch ihren Aufruf bereits eine Wirkung hatten (bzw. einen „Effekt“ zeigten).
- Diese Wirkung wird in der Informatik **Seiteneffekt** genannt.
- Beispiel für einen unerwünschten Seiteneffekt:
 - Eine Funktion gibt auf der Konsole das Ergebnis einer Berechnung aus. Tatsächlich wird dadurch aber die Wiederverwertbarkeit der Funktion eingeschränkt, denn der Programmierer könnte andere Ziele haben:
 - Der Ausgabertext hätte in einer anderen Sprache erfolgen sollen.
 - Der berechnete Wert hätte gar nicht auf der Konsole erscheinen sollen, sondern sollte (z.B.) in einer Datenbank abgespeichert werden.
 - Vor der Weiterverwertung des Rechenergebnisses hätte dieses überprüft werden müssen.
 - ...
- Durch die Einführung von Rückgabewerten kann der Seiteneffekt aber vermieden werden, denn der Programmierer kann nun selbst entscheiden, wie (und ob) der Rückgabewert genutzt wird.

Funktionen(II) – Beispielaufgabe

- Wir werden als Beispiel eine „vollständige“ Funktion betrachten, die mehrere Übergabewerte besitzt, selbstverständlich eine Verarbeitung vornehmen wird, und anschließend (genau) einen Rückgabewert liefert.
- Das Beispiel arbeitet zunächst ausschließlich mit ganzen Zahlen. Die Übergabewerte und der Rückgabewert können aber natürlich auch beliebige andere Typen besitzen – dies werden wir in den Übungsaufgaben erleben.

Aufgabenstellung:

Das Programm startet mit der Abfrage zweier ganzer Zahlen **x** und **y**. Anschließend wird eine Funktion *[siehe unten]* mit den Übergabewerten x und y aufgerufen. Nach dem Abarbeiten der Funktion wird deren Rückgabewert in der Variable **min** gespeichert. Der Wert von *min* wird auf der Konsole ausgegeben und das Programm endet.

Zur Funktion:

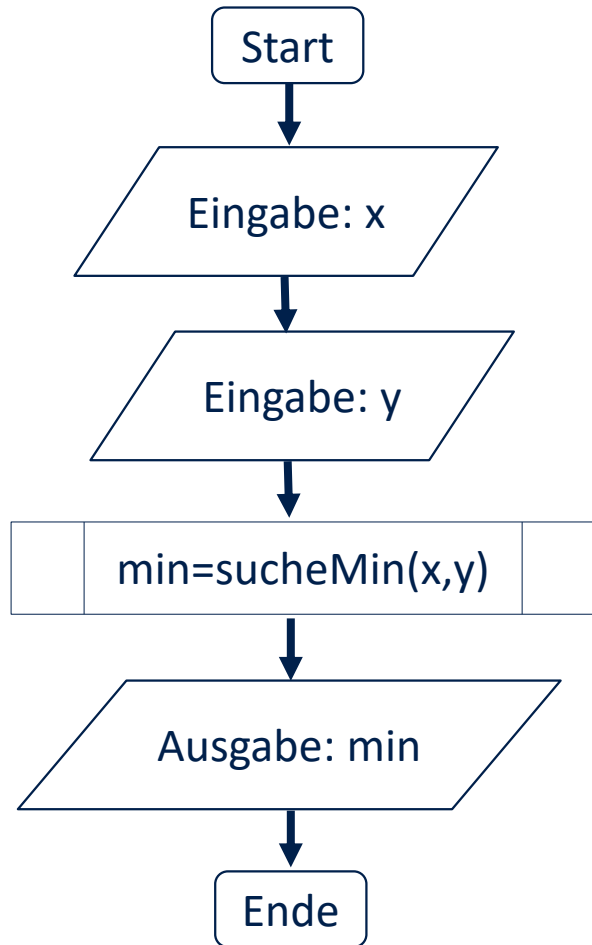
Name: sucheMin

Übergabewerte: 2 Integer a und b

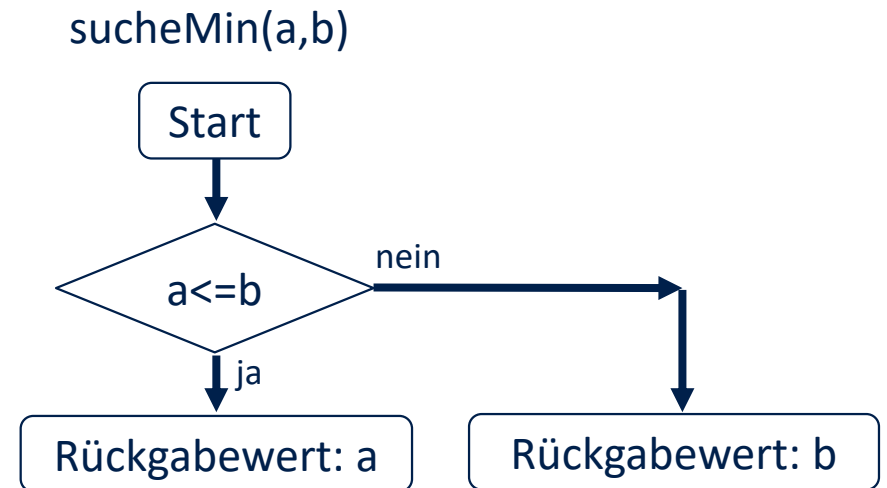
Funktionalität: ermittelt das Minimum aus a und b

Rückgabewert: das ermittelte Minimum

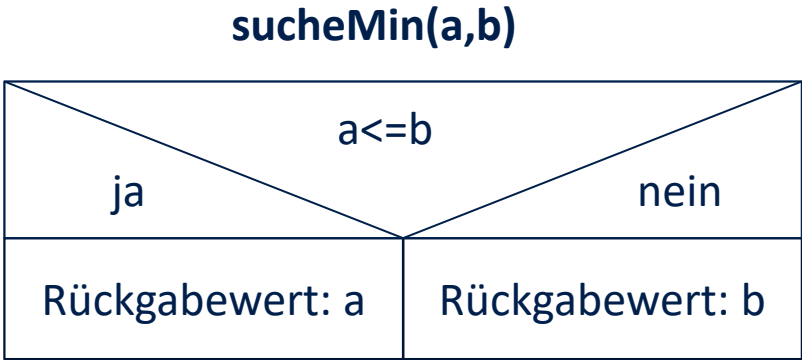
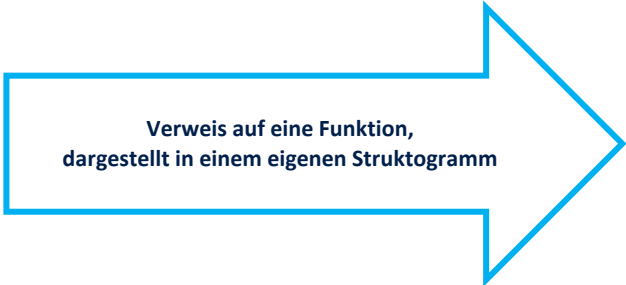
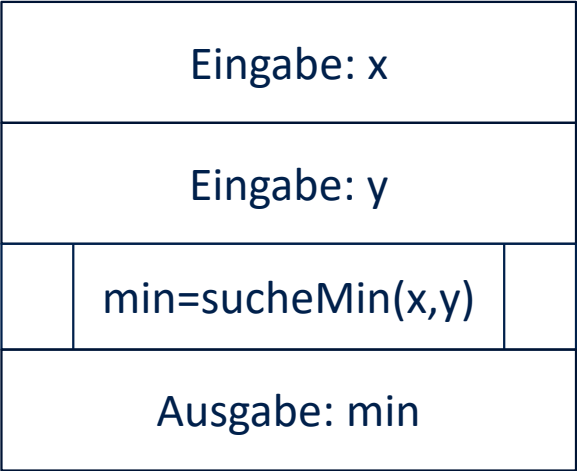
Beispielaufgabe – PAP



Verweis auf eine Funktion,
dargestellt in einem eigenen PAP



Beispielaufgabe – Struktogramm



Beispielaufgabe – Pseudocode

Programm „Beispiel_Rückgabewert“

```
{  
    Eingabe: x  
    Eingabe: y  
    min=sucheMin(x,y)  
    Ausgabe: min  
}
```

Verweis auf eine Funktion

Funktion „sucheMin(a,b)“


```
{  
    falls(a<=b)  
    {  
        return a  
    }  
    sonst  
    {  
        return b  
    }  
}
```


Beispielaufgabe – Quellcode

```
# include<stdio.h>

int sucheMin(int a, int b)
{
    if(a<=b)
    {
        return a;
    }
    else
    {
        return b;
    }
}

main()
{
    int x,y,min;
    printf("Bitte erste Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&x);
    printf("Bitte zweite Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&y);
    min=sucheMin(x,y);
    printf("Das Minimum aus %d und %d ist %d",x,y,min);
}
```




Verweis auf eine Funktion

Beispielaufgabe – Quellcode – Typ des Rückgabewertes

```
# include<stdio.h>

int sucheMin(int a, int b)
{
    if(a<=b)
    {
        return a;
    }
    else
    {
        return b;
    }
}

main()
{
    int x,y,min;
    printf("Bitte erste Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&x);
    printf("Bitte zweite Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&y);
    min=sucheMin(x,y);
    printf("Das Minimum aus %d und %d ist %d",x,y,min);
}
```



Verweis auf eine Funktion

Beispielaufgabe – Quellcode – Rückgabewert des main

```
# include<stdio.h>

int sucheMin(int a, int b)
{
    if(a<=b)
    {
        return a;
    }
    else
    {
        return b;
    }
}

int main()
{
    int x,y,min;
    printf("Bitte erste Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&x);
    printf("Bitte zweite Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&y);
    min=sucheMin(x,y);
    printf("Das Minimum aus %d und %d ist %d",x,y,min);

    return 0;
}
```

Beispielaufgabe – Quellcode – void (bei Rückgabewert)

```
#include<stdio.h>

int sucheMin(int a, int b)
{
    if(a<=b)
    {
        return a;
    }
    else
    {
        return b;
    }
}

void main()
{
    int x,y,min;
    printf("Bitte erste Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&x);
    printf("Bitte zweite Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&y);
    min=sucheMin(x,y);
    printf("Das Minimum aus %d und %d ist %d",x,y,min);

    return 0;
}
```

Beispielaufgabe – Quellcode – void (bei Übergabewert)

```
#include<stdio.h>

int sucheMin(int a, int b)
{
    if(a<=b)
    {
        return a;
    }
    else
    {
        return b;
    }
}

int main(void)
{
    int x,y,min;
    printf("Bitte erste Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&x);
    printf("Bitte zweite Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&y);
    min=sucheMin(x,y);
    printf("Das Minimum aus %d und %d ist %d",x,y,min);

    return 0;
}
```

Beispielaufgabe – Quellcode – void (bei beiden Werten)

```
#include<stdio.h>

int sucheMin(int a, int b)
{
    if(a<=b)
    {
        return a;
    }
    else
    {
        return b;
    }
}

void main(void)
{
    int x,y,min;
    printf("Bitte erste Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&x);
    printf("Bitte zweite Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&y);
    min=sucheMin(x,y);
    printf("Das Minimum aus %d und %d ist %d",x,y,min);

return 0;
}
```

Beispielaufgabe – Quellcode – zukünftige Vorgehensweise

```
# include<stdio.h>

int sucheMin(int a, int b)
{
    if(a<=b)
    {
        return a;
    }
    else
    {
        return b;
    }
}

int main(void)
{
    int x,y,min;
    printf("Bitte erste Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&x);
    printf("Bitte zweite Zahl eingeben: ");
    fflush(stdin);
    scanf("%d",&y);
    min=sucheMin(x,y);
    printf("Das Minimum aus %d und %d ist %d",x,y,min);

    return 0;
}
```

Wir haben bereits gesehen, dass wir bei Funktionen ohne Übergabewert (und/oder Rückgabewert) auf void verzichten könnten.

Es gilt aber als guter Stil, void zu notieren.

Zudem gibt es Programmiersprachen, die die Verwendung von void zwingend vorschreiben und also z.B. eine Schreibweise der Form **main()** nicht zulassen würden.

Zukünftig wollen wir daher wie folgt vorgehen:

- a) Bei **selbsterstellten Funktionen** schreiben wir für fehlende Übergabe- und/oder Rückgabe-Werte in beiden Fällen **void**
- b) Bei der **main-Funktion** verwenden wir **void** für den fehlenden Übergabewert. Als Rückgabewert setzen wir **0** vom Typ **int**.

Funktionen – Gemeinsame Übung A_03_02_01



Aufgabe_03_02_01

Gegeben seien die folgenden Struktogramme:

Hauptprogramm(A_03_02_01)

Eingabe: grundwert

Eingabe: prozentwert

p=prozentsatz(grundwert,prozentwert)

Ausgabe: p

prozentsatz(g,pw)

p=(pw/g)*100

return: p

Aufgabenstellung:

Bitte erstellen Sie dazu einen geeigneten **Quellcode** in ANSI C.

WBS TRAINING AG
Lorenzweg 5
D-12099 Berlin
Amtsgericht Berlin HRB 68531
Sitz der Gesellschaft: Berlin

Vorstand:
Heinrich Kronbichler,
Joachim Giese
Aufsichtsrat (Vorsitz): Dr. Daniel Stadler
USt-IdNr.: DE 209 768 248

GLS Gemeinschaftsbank eG
IBAN: DE18 4306 0967 1146 1814 00
BIC: GENODEM33GLS



ISO certified bank
BIC: GENODEM33GLS
Zulassung nach AGN Reg. Nr. 01/13/0001/2015

**VIELEN DANK
FÜR IHRE
AUFMERKSAMKEIT!**