

Programmierung(1)

Agenda

- **(selbsterzeugte) Funktionen(I)** *(Prozeduren)*
 - Motivation
 - 3 Beispielaufgaben
 - ohne Übergabewerte
 - mit einem Übergabewert
 - mit mehreren Übergabewerten
 - Darstellung in PAP, Struktogramm, Pseudocode
 - Syntax in ANSI C
- Ausführliches Training + Ergebnisbesprechung
- Fachpraktische Anwendungen

(selbsterzeugte) Funktionen(I) *Prozeduren* – Motivation

- Eine wichtige Idee bei der Softwareentwicklung ist das Erstellen von **wiederverwertbarem** Code.
- Auch die uns bereits bekannte Funktion `printf(...)` verfolgt diesen Gedanken, denn sie wurde ein **einziges mal codiert** und mittlerweile schon **millionenfach genutzt**.
- Es handelt sich dabei im Prinzip um kleine „eigenständige“ **Unterprogramme**, die während eines Hauptprogrammes immer dann gestartet werden, wenn deren Funktionalität benötigt wird.
- Da Funktionen von einem Hauptprogramm aus gestartet werden, können wir diese nur testen, wenn wir zusätzlich zu der von uns geschriebenen Funktion auch ein (dann aber in der Regel nur sehr kleines) Hauptprogramm erstellen. (Diesen Gedanken werden wir auch schon bei den Beispielaufgaben berücksichtigen.)

Hinweis:

Wir werden morgen Funktionen kennenlernen, die in ihrem Aufbau dem mathematischen Verständnis dieses Begriffes näher kommen werden. Um den Unterschied deutlich werden zu lassen, könnte man daher die heutigen Beispiele als **Prozeduren** bezeichnen (was in der Programmiersprache „Pascal“ auch tatsächlich so geschieht). In der IT hat es sich aber durchgesetzt, in beiden Fällen von **Funktionen** zu sprechen.

Funktionen (Prozeduren) – Beispielaufgabe(I)

- Als einen ersten Einstieg werden wir zunächst die einfachste Form einer Funktion betrachten.
- Auf diese Weise werden wir kennenlernen, was bei jeder Funktion unverzichtbar ist.
- Zugleich werden wir nachvollziehen können, warum wir dieses Konzept noch ausbauen wollen, denn die in diesem ersten Beispiel vorgestellte Funktion ist in gewisser Hinsicht „wenig intelligent“, da sie weder interaktiv ist, noch etwas berechnet.

Aufgabenstellung:

Das Programm startet mit einer Ausgabe: „Gleich wird eine Funktion aufgerufen:“

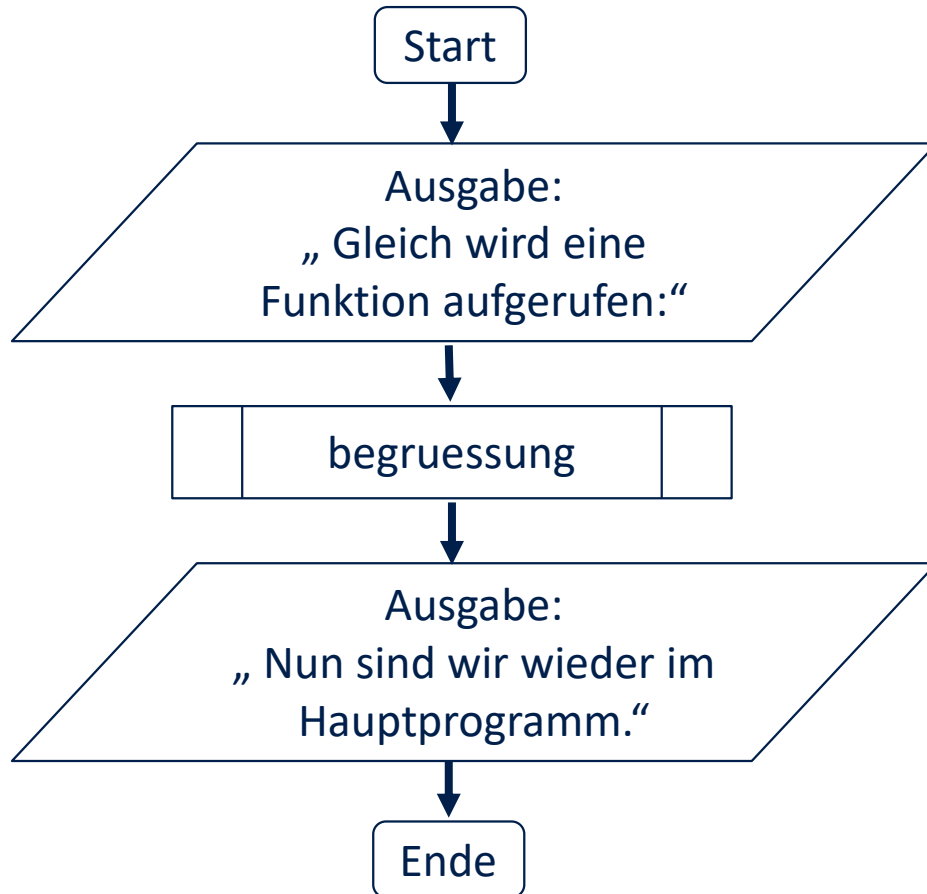
Anschließend wird die Funktion „begruessung“ gestartet.

Diese Funktion hat nur eine einzige Aufgabe, nämlich die Ausgabe von „Hallo zusammen!“

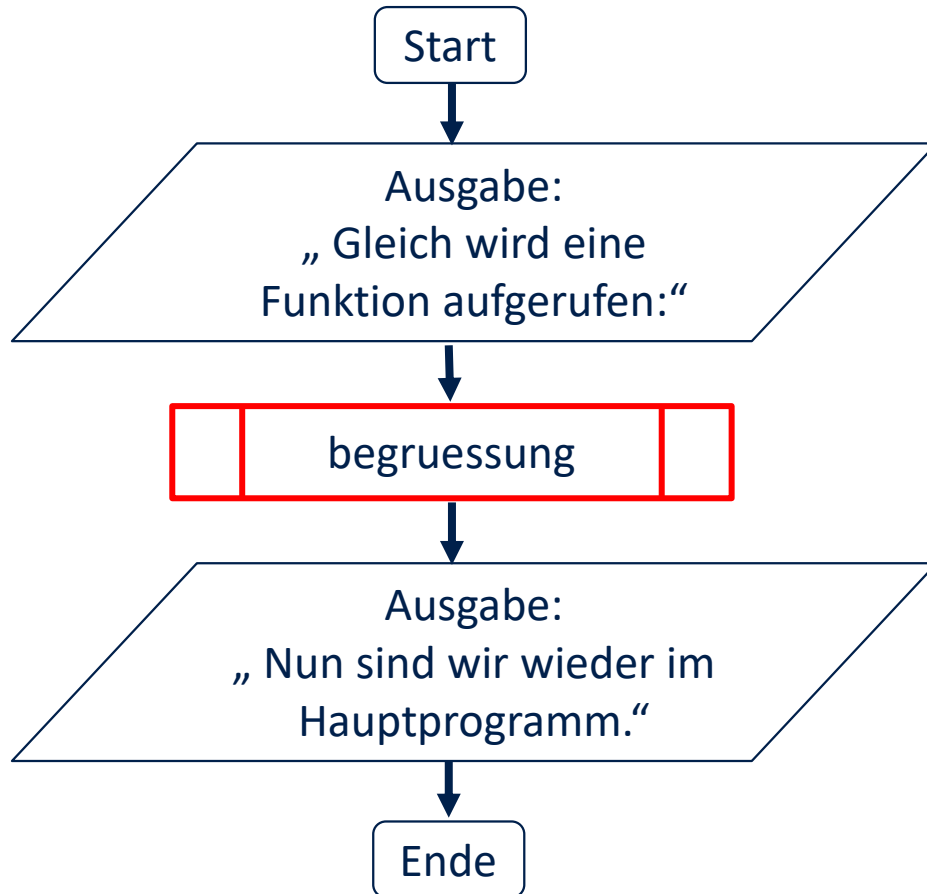
Nachdem diese Funktion abgearbeitet wurde, geht es im Hauptprogramm weiter:

Es folgt die Ausgabe: „Nun sind wir wieder im Hauptprogramm.“ und das Programm endet.

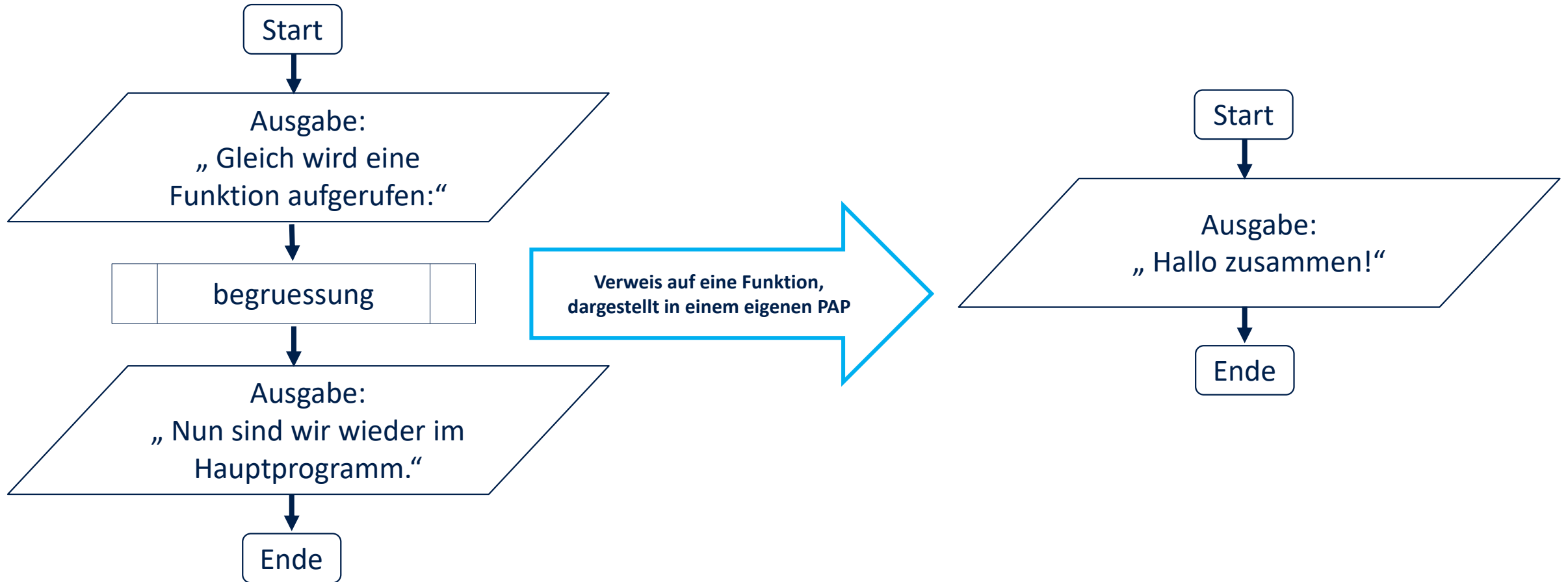
Beispielaufgabe – PAP



Beispielaufgabe – PAP – Symbol für Funktionen

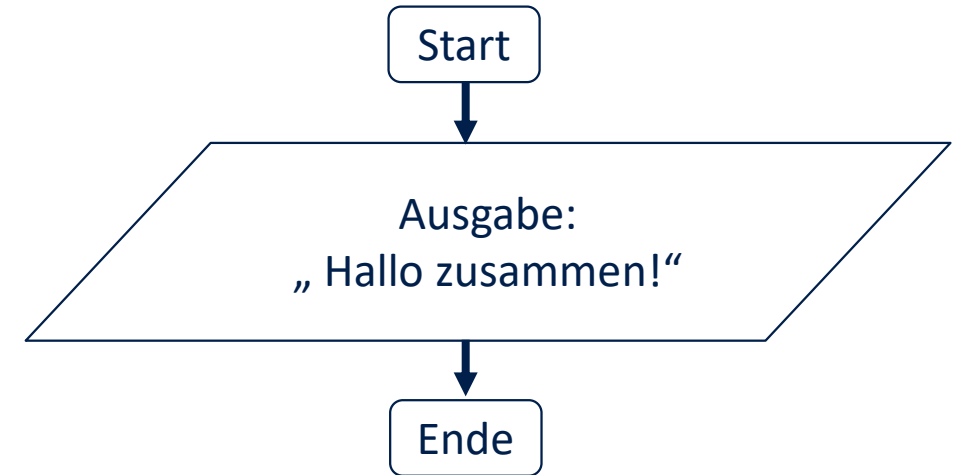
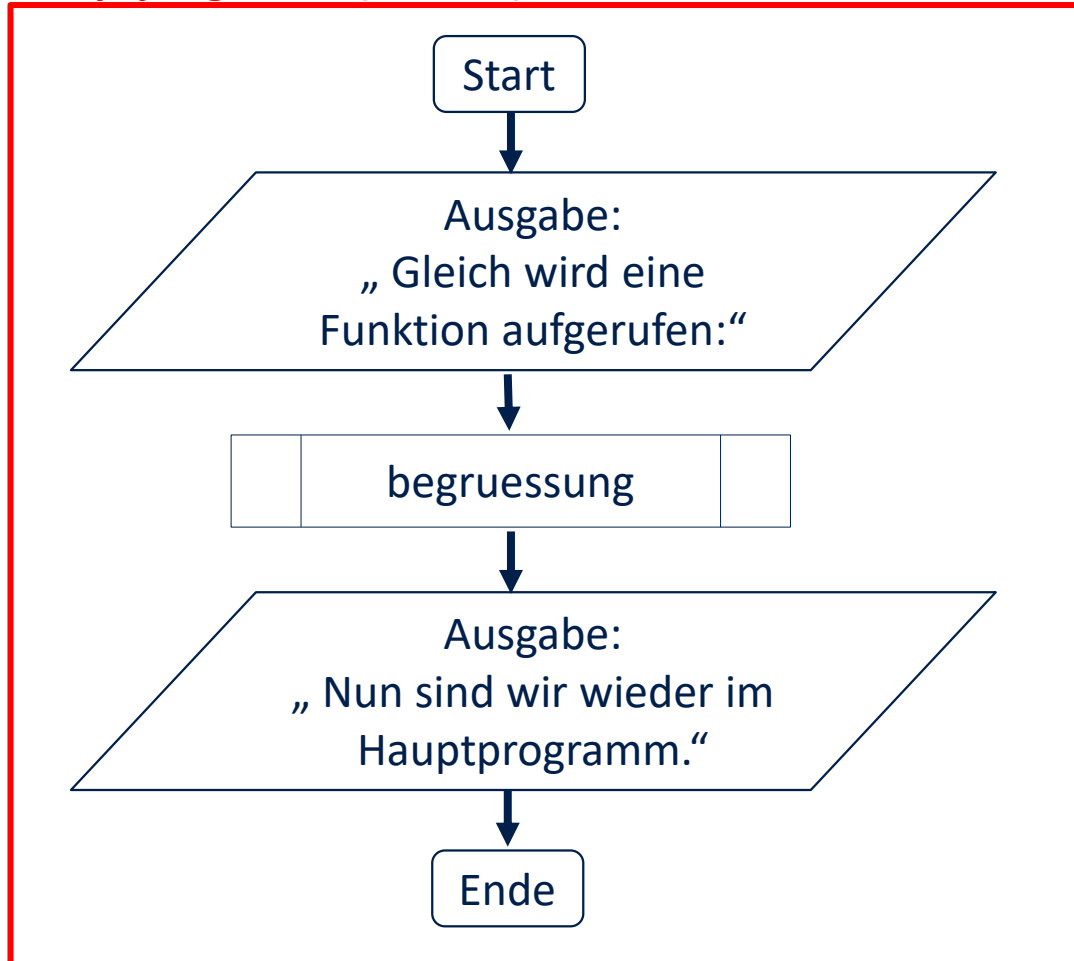


Beispielaufgabe – PAP

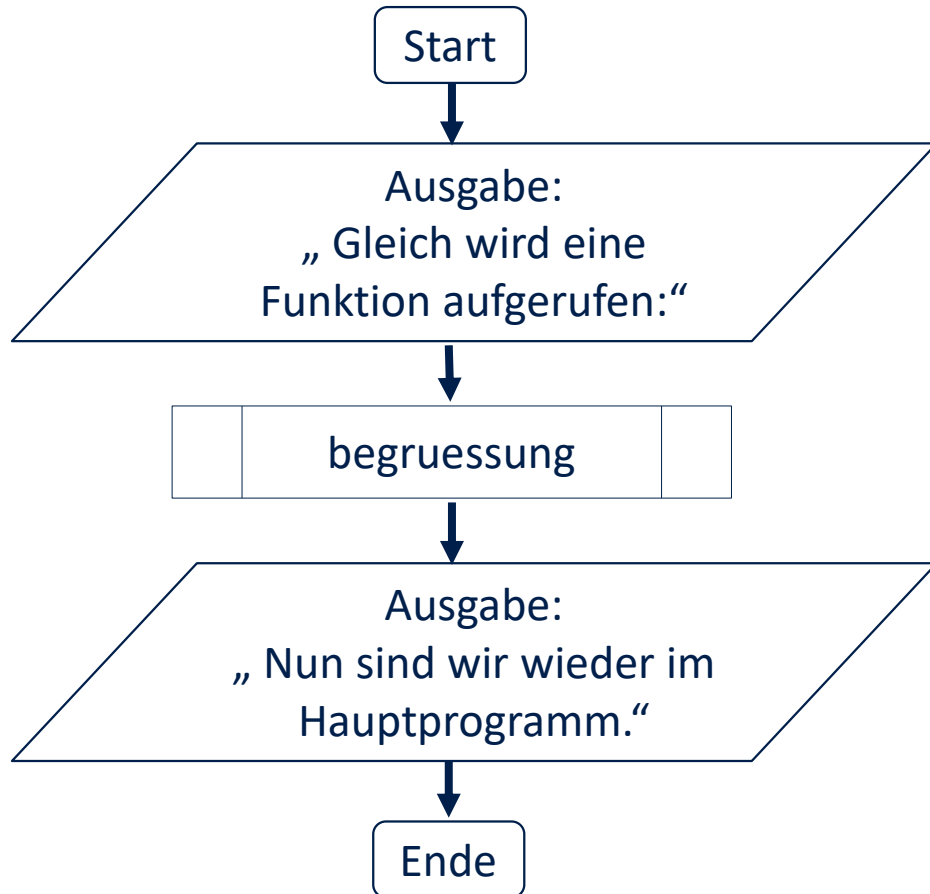


Beispielaufgabe – PAP

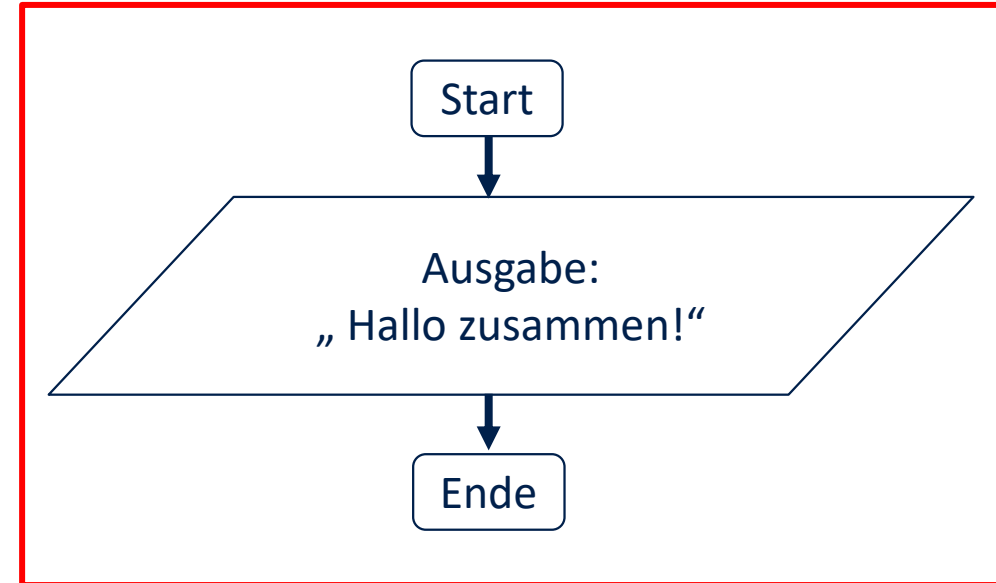
Hauptprogramm („main“)



Beispielaufgabe – PAP



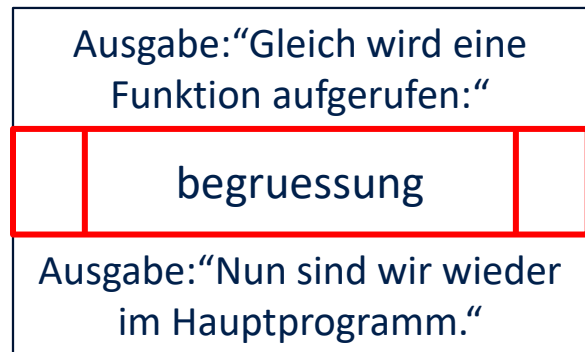
Funktion („begrueessung“)



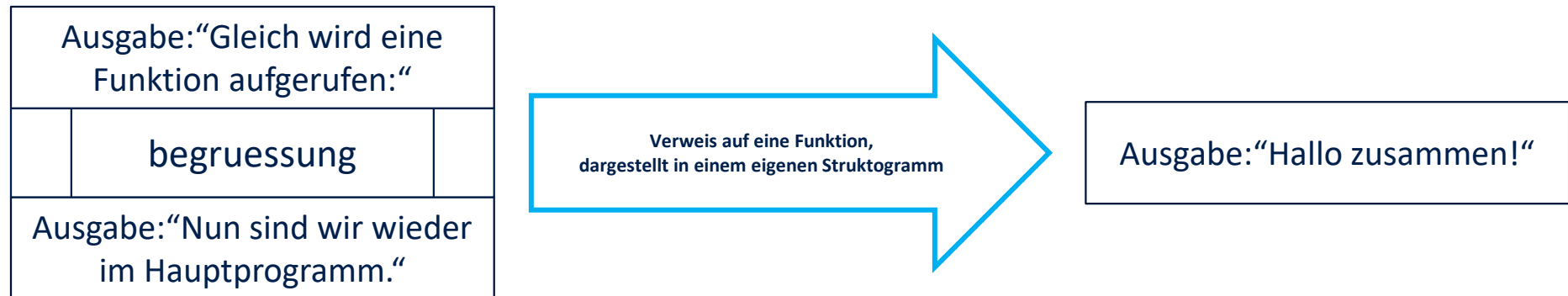
Beispielaufgabe – Struktogramm

Ausgabe: "Gleich wird eine Funktion aufgerufen:"		
	begrueessung	
Ausgabe: "Nun sind wir wieder im Hauptprogramm."		

Beispielaufgabe – Struktogramm – Symbol für Funktionen

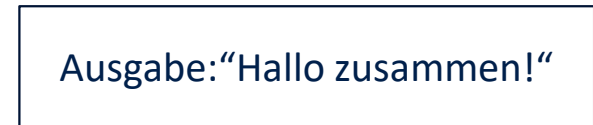
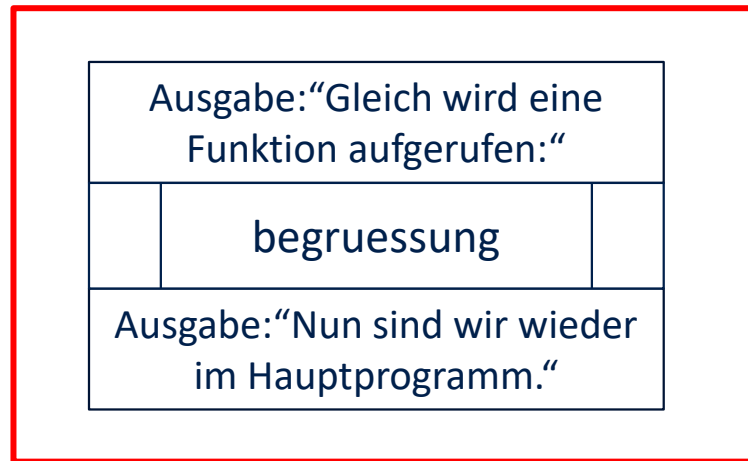


Beispielaufgabe – Struktogramm

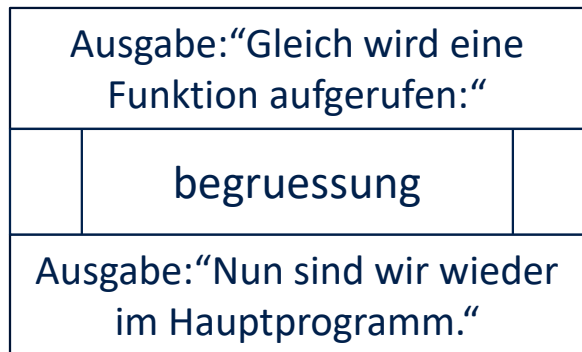


Beispielaufgabe – Struktogramm

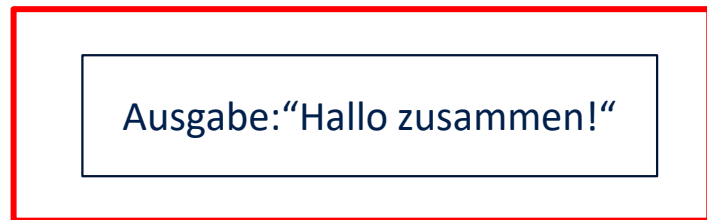
Hauptprogramm („main“)



Beispielaufgabe – Struktogramm



Funktion („begrueessung“)



Beispielaufgabe – Pseudocode

Programm „Beispiel 1“

```
{  
    Ausgabe: "Gleich wird eine Funktion aufgerufen:"  
    begruessung()  
    Ausgabe: "Nun sind wir wieder im Hauptprogramm."  
}
```

Beispielaufgabe – Pseudocode – Schreibweise für Funktionen

Programm „Beispiel 1“

```
{  
    Ausgabe: "Gleich wird eine Funktion aufgerufen:"  
    begrüessung()  
    Ausgabe: "Nun sind wir wieder im Hauptprogramm."  
}
```


Beispielaufgabe – Pseudocode

Programm „Beispiel 1“

{

Ausgabe: „Gleich wird eine Funktion aufgerufen:“

begruessung()

Verweis auf eine Funktion

Ausgabe: „Nun sind wir wieder im Hauptprogramm.“

}

Funktion „begruessung“

{

Ausgabe: „Hallo zusammen!“

}

Beispielaufgabe – Quellcode

```
# include<stdio.h>

main()
{
    printf("Gleich wird eine Funktion aufgerufen:");
    begruessung();
    printf("Nun sind wir wieder im Hauptprogramm.");
}
```

Beispielaufgabe – Quellcode – Schreibweise für Funktionen

```
# include<stdio.h>


main()
{
    printf("Gleich wird eine Funktion aufgerufen:");
    begruessung();
    printf("Nun sind wir wieder im Hauptprogramm.");
}
```

Beispielaufgabe – Quellcode

```
# include<stdio.h>

begruessung()
{
    printf("Hallo zusammen!");
}

main()
{
    printf("Gleich wird eine Funktion aufgerufen:");
    begruessung();
    printf("Nun sind wir wieder im Hauptprogramm.");
}
```



Verweis auf eine Funktion

Beispielaufgabe – Quellcode

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

```
begruessung()  
{  
    printf("Hallo zusammen!");  
}
```

Hauptprogramm („main“)

```
main()  
{  
    printf("Gleich wird eine Funktion aufgerufen:");  
    begruessung();  
    printf("Nun sind wir wieder im Hauptprogramm.");  
}
```

Beispielaufgabe – Quellcode

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

Funktion („begruessung“)

```
begruessung()  
{  
    printf("Hallo zusammen!");  
}
```

```
main()  
{  
    printf("Gleich wird eine Funktion aufgerufen:");  
    begruessung();  
    printf("Nun sind wir wieder im Hauptprogramm.");  
}
```

Funktionen (Prozeduren) – Beispielaufgabe(II)

- Beim zweiten Beispiel wollen wir eine „intelligenterere“ Funktion vorstellen.
- Gemeint ist: Die Funktion wird nicht stets das selbe leisten, sondern auf einen „Input“ reagieren.
- Dieser Input wird als **Übergabewert** (oder auch: **Parameter**) bezeichnet.

Aufgabenstellung:

Das Programm fragt zu Beginn den Lieblingsbuchstaben des Users ab.

Anschließend wird die Funktion „meinLiebling“ gestartet.

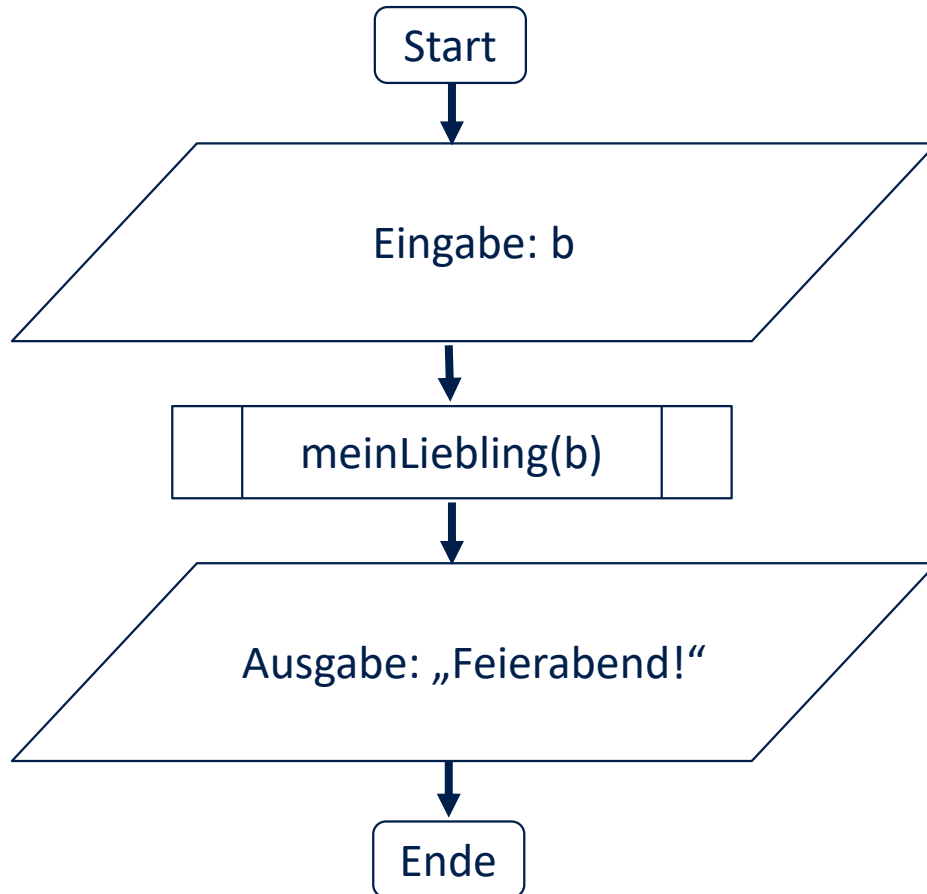
Als Übergabewert wird der Funktion der vom User eingegebene Buchstabe übergeben.

Die Funktion meinLiebling arbeitet eine Schleife ab, um den Übergabewert 10-mal auf der Konsole auszugeben.

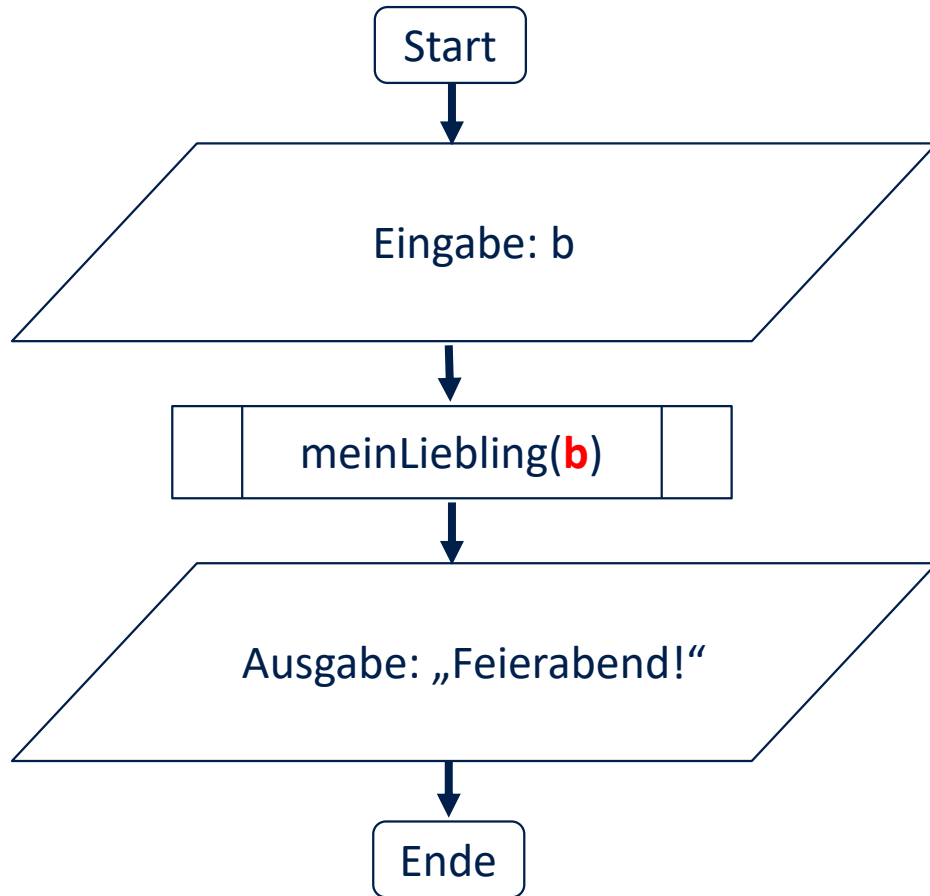
Nachdem die Funktion beendet wurde, geht es im Hauptprogramm weiter:

Es folgt die Ausgabe: „Feierabend!“ und das Programm endet.

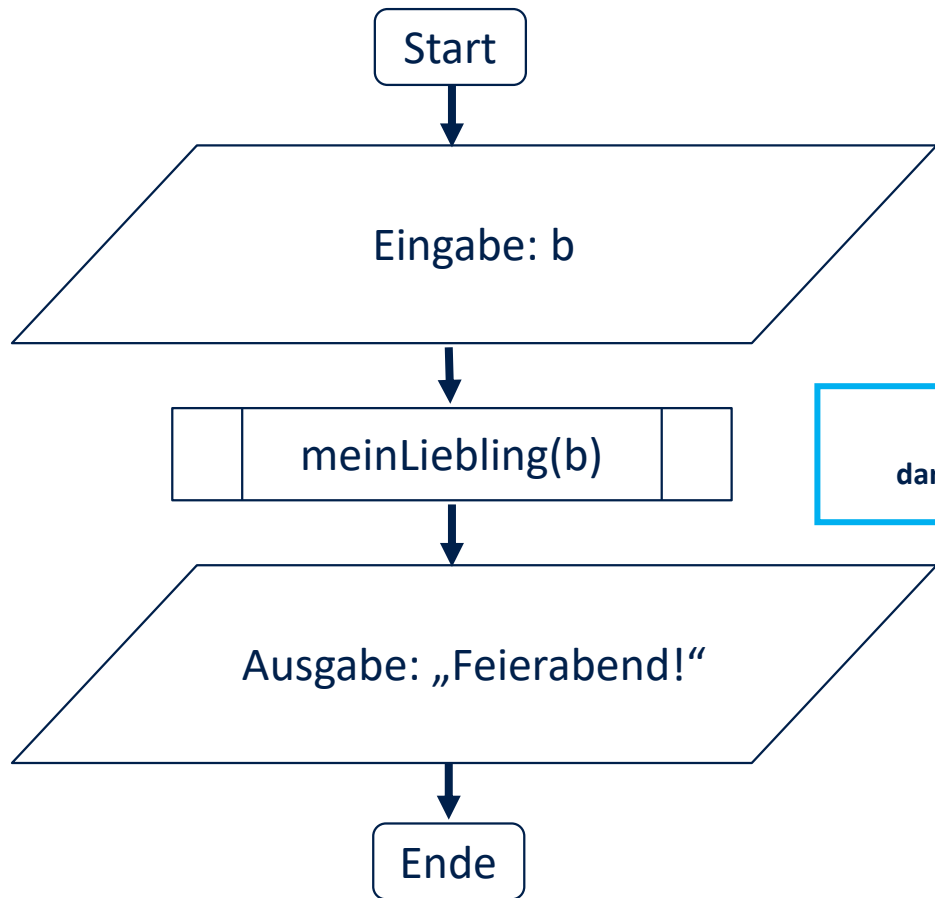
Beispielaufgabe – PAP



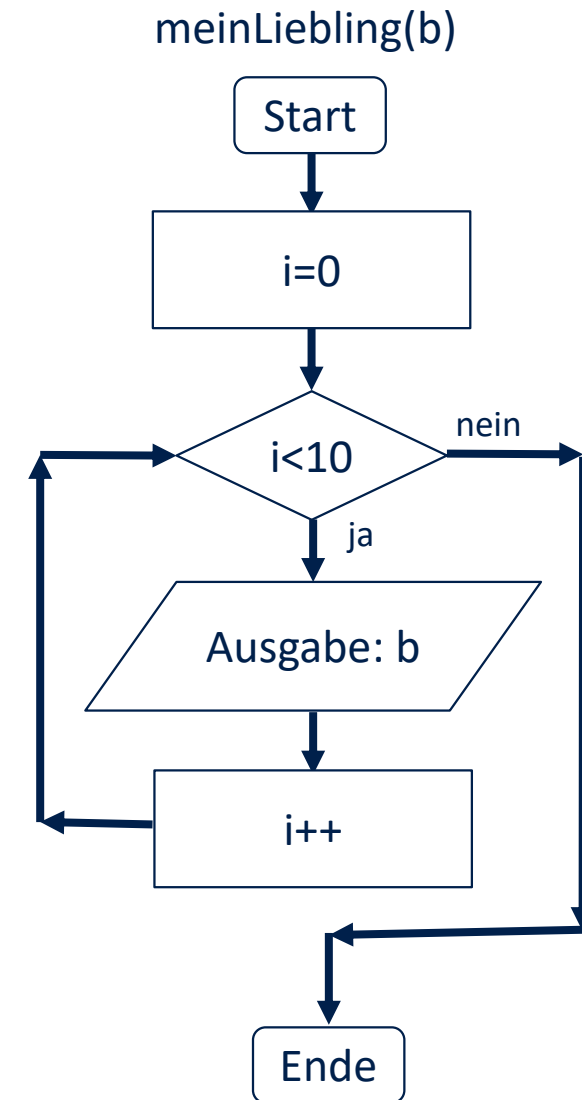
Beispielaufgabe – PAP – Übergabewert



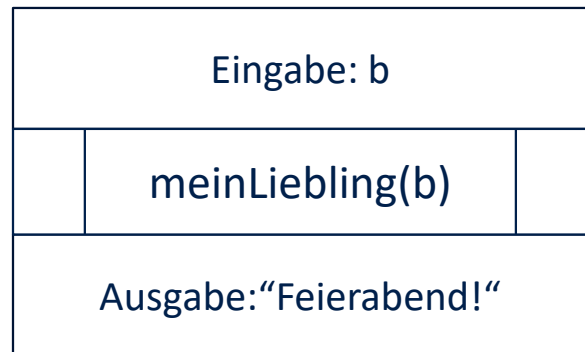
Beispielaufgabe – PAP



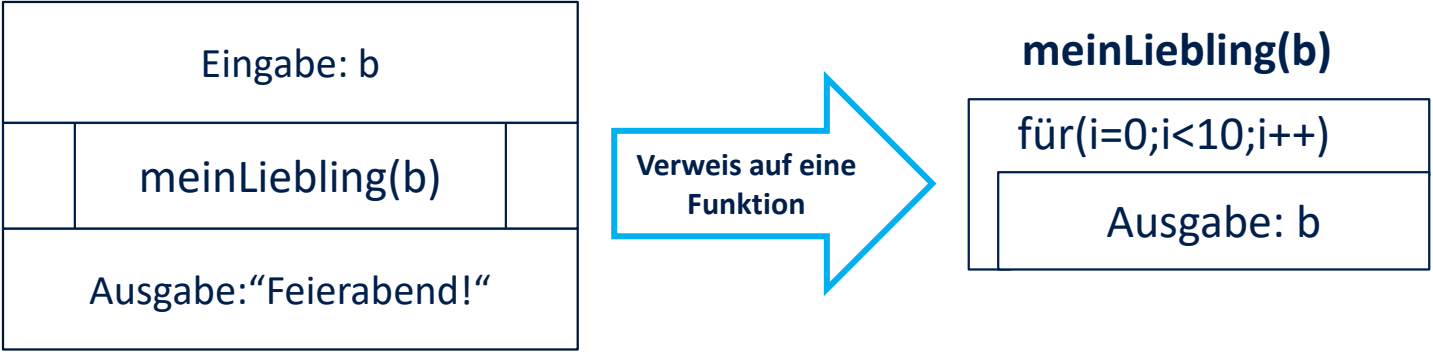
Verweis auf eine Funktion,
dargestellt in einem eigenen PAP



Beispielaufgabe – Struktogramm



Beispielaufgabe – Struktogramm



Beispielaufgabe – Pseudocode

Programm „Beispiel 2“

```
{  
    Eingabe: b  
    meinLiebling(b)  
    Ausgabe: "Feierabend!"  
}
```

Beispielaufgabe – Pseudocode

Programm „Beispiel 2“

{

Eingabe: b

meinLiebling(b)

Ausgabe: "Feierabend!"

}

Verweis auf eine Funktion

meinLiebling(b)

{

für(i=0;i<10;i++)

{

Ausgabe: b

}

}

Beispielaufgabe – Quellcode

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

```
main()  
{
```

```
    char b;  
    printf("Geben Sie bitte Ihren Lieblingsbuchstaben ein: ");  
    scanf("%c",&b);  
    meinLiebling(b);  
    printf("Feierabend!");
```


```
}
```

Beispielaufgabe – Quellcode

```
#include<stdio.h>

meinLiebling(char b)
{
    int i;
    for(i=0;i<10;i++)
    {
        printf("%c",b);
    }
}

main()
{
    char b;
    printf("Geben Sie bitte Ihren Lieblingsbuchstaben ein: ");
    scanf("%c",&b);
    meinLiebling(b);
    printf("Feierabend!");
}
```



Verweis auf eine Funktion

Beispielaufgabe – Quellcode – Deklaration des Übergabewertes

```
#include<stdio.h>

meinLiebling(char b)
{
    int i;
    for(i=0;i<10;i++)
    {
        printf("%c",b);
    }
}

main()
{
    char b;
    printf("Geben Sie bitte Ihren Lieblingsbuchstaben ein: ");
    scanf("%c",&b);
    meinLiebling(b);
    printf("Feierabend!");
}
```

Beispielaufgabe – Quellcode – Gültigkeit von Variablen

```
# include<stdio.h>

meinLiebling(char b)
{
    int i;
    for(i=0;i<10;i++)
    {
        printf("%c",b);
    }
}

main()
{
    char b;
    printf("Geben Sie bitte Ihren Lieblingsbuchstaben ein: ");
    scanf("%c",&b);
    meinLiebling(b);
    printf("Feierabend!");
}
```

Funktionen (Prozeduren) – Beispielaufgabe(III)

- Beim dritten Beispiel wollen wir zeigen, wie man mit mehreren Übergabewerten vorgeht.
- Hierfür werden wir das vorangegangene Beispiel aufgreifen und ausbauen.
- Anzahl und Typen der Übergabewerte sind beliebig, im Beispiel reichen uns 2 Parameter.

Aufgabenstellung:

Das Programm fragt zu Beginn erneut den Lieblingsbuchstaben des Users ab.

Daraufhin wird aber auch abgefragt, wie oft dieser Wert ausgegeben werden soll.

Anschließend wird die Funktion „x_mal_meinZeichen“ gestartet.

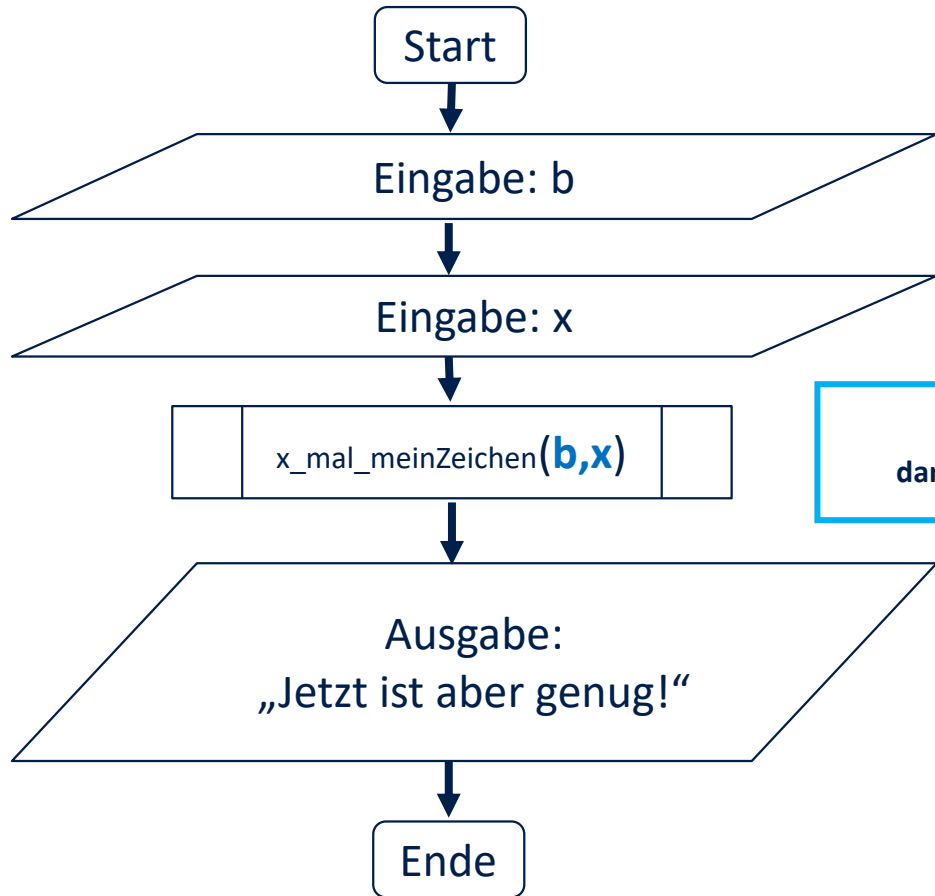
Als Übergabewerte werden der Funktion Buchstabe und Anzahl übergeben.

Die Funktion x_mal_meinZeichen arbeitet eine Schleife ab, um den Buchstaben x-mal auf der Konsole auszugeben.

Nachdem die Funktion beendet wurde, geht es im Hauptprogramm weiter:

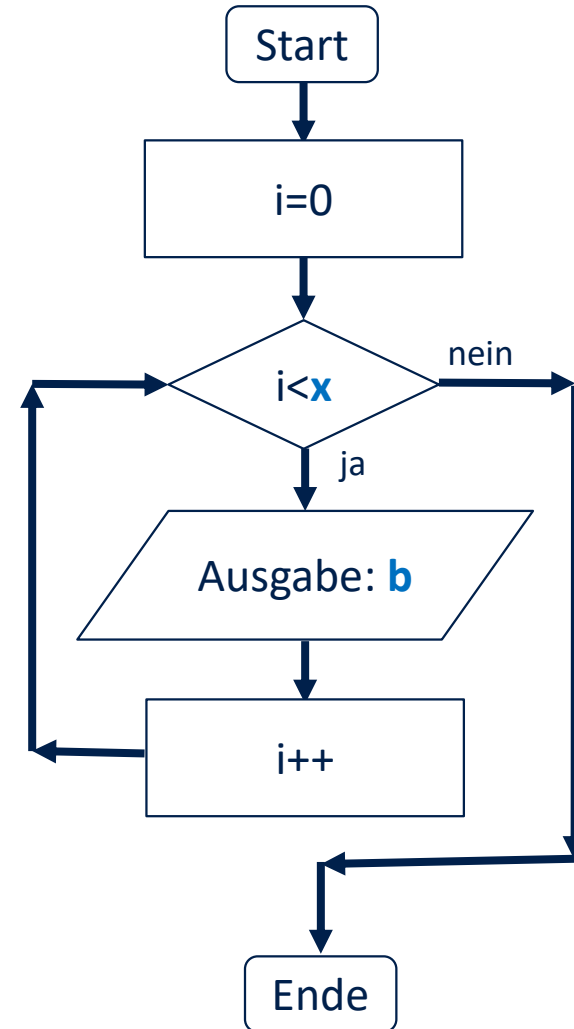
Es folgt die Ausgabe: „Jetzt ist aber genug!“ und das Programm endet.

Beispielaufgabe – PAP

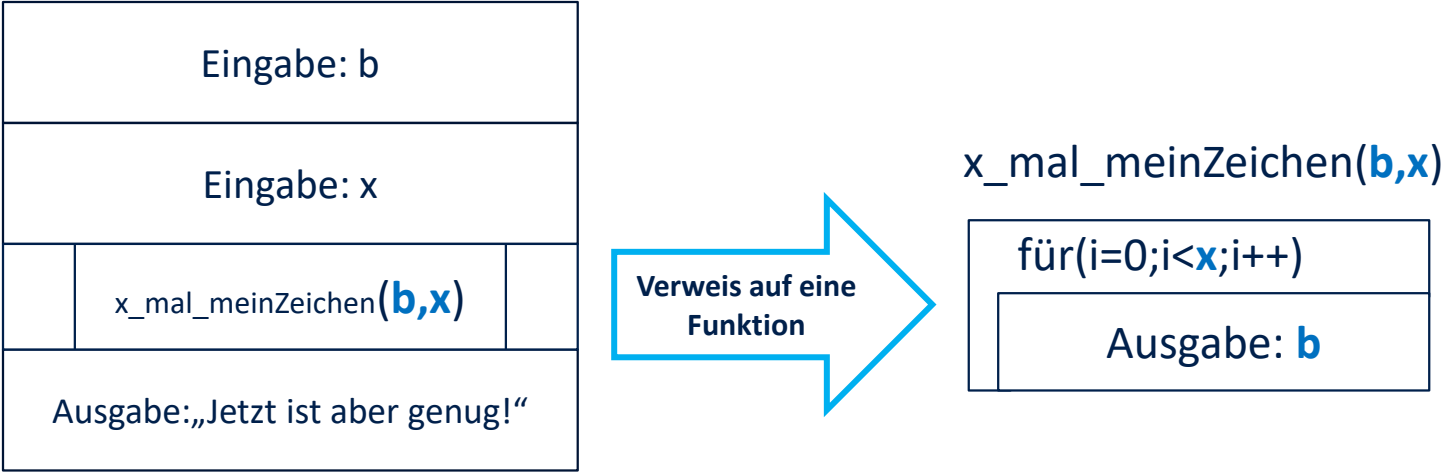


Verweis auf eine Funktion,
dargestellt in einem eigenen PAP

`x_mal_meinZeichen(b,x)`



Beispielaufgabe – Struktogramm



Beispielaufgabe – Pseudocode

Programm „Beispiel 3“

{

 Eingabe: b

 Eingabe: x

 x_mal_meinZeichen (b,x)

 Ausgabe : „Jetzt ist aber genug!“

}



x_mal_meinZeichen(b,x)

{

 für(i=0;i<x;i++)

 {

 Ausgabe: b

 }

}

Beispielaufgabe – Quellcode

```
# include<stdio.h>

x_mal_meinZeichen(char b, int x)
{
    int i;
    for(i=0;i<x;i++)
    {
        printf("%c",b);
    }
}

main()
{
    char b;
    int x;
    printf("Geben Sie bitte Ihren Lieblingsbuchstaben ein: ");
    scanf("%c",&b);
    printf("Geben Sie bitte die gewünschte Anzahl ein: ");
    fflush(stdin);
    scanf("%d",&x);
    x_mal_meinZeichen(b,x);
    printf("Jetzt ist aber genug!");
}
```

Im Rahmen der Definition einer selbsterstellten Funktion wird der **Typ** und die **Reihenfolge** aller Parameter festgelegt.

Verweis auf eine Funktion

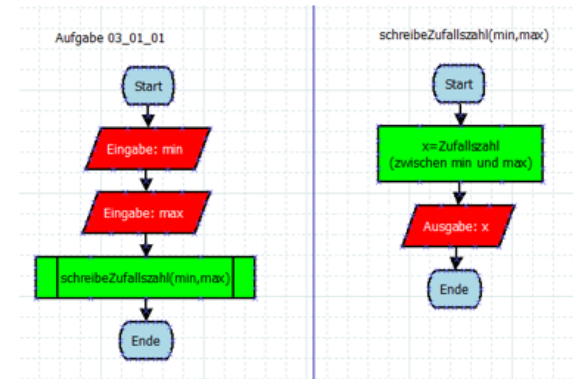
Beim Aufruf dieser Funktion müssen dann aber natürlich Typ und Reihenfolge dieser Parameter beachtet werden.

Funktionen – Gemeinsame Übung A_03_01_01



Aufgabe_03_01_01

Gegeben sei der folgende PAP:



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: GENODEM3GLS



GLS zertifiziert nach
DIN EN ISO 9001:2015 Reg. Nr. 0110040128103
Zulassung nach AGN Reg. Nr. 0110040128103

**VIELEN DANK
FÜR IHRE
AUFMERKSAMKEIT!**