

Java Schleifen: **for**

1. Zählergesteuerte for-Schleife

bedeutet: Wir haben eine **Zählervariable** („**Laufvariable**“), die wir zum Zählen benutzen.

Der Computer „merkt“ sich anhand dieser Variablen, wie weit er gerade ist.

counter = 0

“Hey, wo bist du gerade?” – „Easy, ich bin bei 0“

counter = 1

“Hey, wo bist du gerade?” – „Easy, ich bin bei 1“

counter = 2

“Hey, wo bist du gerade?” – „Easy, ich bin bei 2“

Schleife ("for ..." / "while ...")

Zählergesteuerte Schleife ("for")

*Abbruchkriterium:
Zählvariable >/< Endwert*

zähle Variable von Startwert bis Endwert in
Schrittweite x

Anweisung

noch eine Anweisung ...

Beispiel

zähle x:int von 1 bis 10, Schrittweite 1

Ausgabe: x

Ausgabe: "Cool, was?" + Zeilenumbruch

1a. Zählergesteuerte for-Schleife in Java programmieren und im Struktogramm darstellen

Syntax von for:

```
for (Startwert; Bedingung; Zähler)
{
    Anweisung
}
```

Beispiel: Zahlen von 0 bis 9 ausgeben

```
for (int i = 0; i < 10; i++)
{
    System.out.println(i);
}
```



Syntax von for:

```
for (Startwert; Bedingung; Zähler)
{
    Anweisung
}
```

Beispiel: Zahlen von 0 bis 9 ausgeben

```
for (int i = 0; i < 10; i++)
{
    System.out.println(i);
}
```

lokale Variable

definiert Startwert (= 0)

Syntax von for:

```
for (Startwert; Bedingung; Zähler)
{
    Anweisung
}
```

Beispiel: Zahlen von 0 bis 100 ausgeben

```
for (int i = 0; i <= 100; i++)
{
    System.out.println(i);
}
```

Abbruchbedingung

**(so lange durchführen, wie
i kleiner/gleich 100 ist)**

Syntax von for:

```
for (Startwert; Bedingung; Zähler)
{
    Anweisung
}
```

Beispiel: Zahlen von 0 bis 100 ausgeben

```
for (int i = 0; i <= 100; i++)
{
    System.out.println(i);
}
```

Zähler inkrementieren

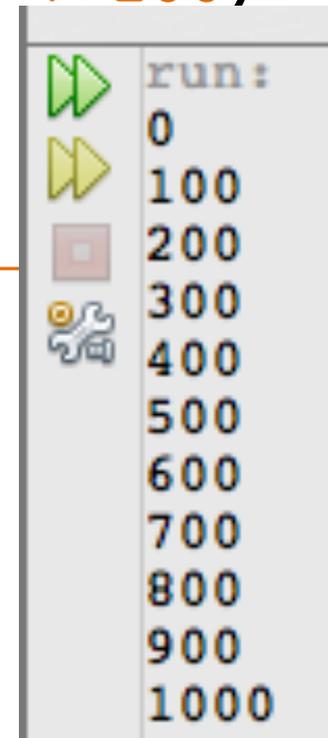
i++ entspricht i = i+1

i-- entspricht i = i-1



Beispiel: Zahlen von 0 bis 1000 in Hunderter- schritten ausgeben

```
for (int i = 0; i <= 1000; i = i + 100)
{
    System.out.println(i);
}
```



```
run:
0
100
200
300
400
500
600
700
800
900
1000
```

for-Schleife im Struktogramm

Beispiel: Zahlen von 0 bis 9 ausgeben

```
for (int i = 0; i < 10; i++)  
{  
    System.out.println(i);  
}
```

+hochzaehlen():void

zähle i:int von 0 bis 9, Schrittweite 1

Ausgabe: i + Zeilenumbruch

for-Schleife im Struktogramm

Der Spruch lautet immer:

Zähle *lokale Variable* von *anfangswert*
bis *endwert*, Schrittweite *abstand*

```
+hochzaehlen():void
```

```
zähle i:int von 0 bis 9, Schrittweite 1
```

```
Ausgabe: i + Zeilenumbruch
```

Übung 1: Einfache for-Schleife

Wir wollen die Zahlen von 10 bis 1000 ausgeben. Dazu benutzen wir eine Methode „zaehlen1()“

- a) Erstellen Sie ein Struktogramm.
- b) Programmieren Sie die Methode.

Übung 1: Einfache for-Schleife

LÖSUNG

Wir wollen die Zahlen von 10 bis 1000 ausgeben. Dazu benutzen wir eine Methode „zaehlen1()“

- Erstellen Sie ein Struktogramm.
- Programmieren Sie die Methode.

+zaehlen1():void

zähle b:int von 10 bis 1000, Schrittweite 1

Ausgabe: b + Zeilenumbruch

```
public void zaehlen1() {  
    for(int b = 10; b <= 1000; b++) {  
        System.out.println(b);  
        // oder  
        // System.out.print(b + "\n");  
    }  
}
```

Übung 1b: Einfache for-Schleife mit Parameter

Zählen Sie in Zweiserschritten. Start- und Endwert werden der Methode als Parameter übergeben. Die Laufvariable wird ausgegeben.

- a) Erstellen Sie ein Struktogramm.
- b) Programmieren Sie die Methode.

Übung 1b: Einfache for-Schleife mit Parameter

LÖSUNG

Zählen Sie in Zweiserschritten. Start- und Endwert werden der Methode als Parameter übergeben.

Die Laufvariable wird ausgegeben.

- Erstellen Sie ein Struktogramm.
- Programmieren Sie die Methode.

+zaehlen2(startwert:int, endwert:int):void

zähle i:int von startwert bis endwert,
Schrittweite 2

Ausgabe: i + Zeilenumbruch

```
public void zaehlen2(int startwert, int endwert) {  
    for(int i = startwert; i <= endwert; i = i + 2) {  
        System.out.println(i);  
    }  
}
```

Übung 1c: Einfache for-Schleife

Geben Sie die Zahlen von 27 bis 18256 in Dreierschritten aus.

Übung 1c: Einfache for-Schleife

LÖSUNG

Geben Sie die Zahlen von 27 bis 18256 in Dreierschritten aus.

```
public void krummeZahlenAusgeben() {  
    for(int brpf = 27;brpf <= 18256;brpf = brpf + 3) {  
        // brpf könnte natürlich auch zahl heißen  
        // oder krummeZahl oder ... oder ...  
        System.out.println(brpf);  
    }  
}
```

Übung 1d: Struktogramm zu einfacher for-Schleife

Erstellen Sie zu dieser Methode ein Struktogramm:

```
public void krummeZahlenAusgeben() {  
    for(int brpf = 27;brpf <= 18256;brpf = brpf + 3) {  
        // brpf könnte natürlich auch zahl heißen  
        // oder krummeZahl oder ... oder ...  
        System.out.println(brpf);  
    }  
}
```

Übung 1d: Struktogramm zu einfacher for-Schleife

LÖSUNG

```
public void krummeZahlenAusgeben() {  
    for(int brpf = 27;brpf <= 18256;brpf = brpf + 3) {  
        // brpf könnte natürlich auch zahl heißen  
        // oder krummeZahl oder ... oder ...  
        System.out.println(brpf);  
    }  
}
```

+krummeZahlenAusgeben():void

zähle brpf:int von 27 bis 18256, Schrittweite 3

Ausgabe: brpf + Zeilenumbruch

Übung 1e: Struktogramm zu einfacher for-Schleife

Erstellen Sie zu dieser Methode ein Struktogramm:

```
public void zahlenVon0BisParameterAusgeben(int endwert) {  
    for(int zahl = 0; zahl <= endwert; zahl++) {  
        System.out.println(zahl);  
    }  
}
```

Übung 1e: Struktogramm zu einfacher for-Schleife

LÖSUNG

```
public void zahlenVon0BisParameterAusgeben(int endwert) {  
    for(int zahl = 0; zahl <= endwert; zahl++) {  
        System.out.println(zahl);  
    }  
}
```

+zahlenVon0BisParameterAusgeben(endwert:int):void

zähle zahl:int von 0 bis endwert, Schrittweite 1

Ausgabe: zahl + Zeilenumbruch

Übung 1f: Struktogramm zu einfacher for-Schleife

Erstellen Sie zu dieser Methode ein Struktogramm:

```
public void etwasTun() {  
    int b = 1350;  
    int x = 1820;  
    for(int i = b; i <= x; i++) {  
        int erg = i + x;  
        System.out.println("x ist " + x);  
        System.out.println("erg ist " + erg);  
    }  
}
```

Übung 1f: Struktogramm zu einfacher for-Schleife

LÖSUNG

```
public void etwasTun() {  
    int b = 1350;  
    int x = 1820;  
    for(int i = b; i <= x; i++) {  
        int erg = i + x;  
        System.out.println("x ist " + x);  
        System.out.println("erg ist " + erg);  
    }  
}
```

+etwasTun():void

b:int = 1350

x:int = 1820

zähle von i:int = b bis x, Schrittweite 1

erg:int = i + x + Zeilenumbruch

Ausgabe: "x ist " + x + Zeilenumbruch

Ausgabe: "erg ist " + erg + Zeilenumbruch

2. foreach-Schleifen

foreach-Schleife

```
private ArrayList<String> namensliste;  
public Schleife() {  
    this.namensliste = new ArrayList<String>();  
    this.namensliste.add("Chantal");  
    this.namensliste.add("Susanne");  
    this.namensliste.add("Heinrich");  
}  
public void durchlaufen() {  
    for (String einName : this.namensliste) {  
        System.out.println(einName);  
    }  
}
```

foreach-Schleife

```
private ArrayList<String> namensliste;  
public Schleife() {  
    this.namensliste = new ArrayList<String>();  
    this.namensliste.add("Chantal");  
    this.namensliste.add("Susanne");  
    this.namensliste.add("Heinrich");  
}
```

```
public void durchlaufen() {  
    for (String einName : this.namensliste) {  
        System.out.println(einName);  
    }  
}
```

+durchlaufen():void

foreach(einName:String : this.namensliste

Ausgabe: einName + Zeilenumbruch

foreach-Schleife

```
private ArrayList<String> namensliste;  
public Schleife() {  
    this.namensliste = new ArrayList<String>();  
    this.namensliste.add("Chantal");  
    this.namensliste.add("Susanne");  
    this.namensliste.add("Heinrich");  
}
```

```
public void durchlaufen() {  
    for (String einName : this.namensliste) {  
        System.out.println(einName);  
    }  
}
```

+durchlaufen():void

foreach **einName:String** : **this.namensliste**

Ausgabe: einName + Zeilenumbruch

foreach-Schleife

```
foreach(einName:String : this.namensliste
```

```
ne:String : this.na
```

keine Leerzeichen

Leerzeichen

Übung 2a: foreach-Schleife nach Struktogramm programmieren

Setzen Sie dieses Struktogramm programmiertechnisch um.

+testen(a:int):void

lokaleListe:ArrayList<Integer> = new ArrayList<Integer>()

17 zu lokaleListe hinzufügen

32 zu lokaleListe hinzufügen

11 zu lokaleListe hinzufügen

foreach(eineZahl:int : lokaleListe)

J

eineZahl < a

N

Ausgabe: "Treffer!" + Zeilenumbruch

Ausgabe: eineZahl + " - leider kein Treffer."
+ Zeilenumbruch

Ausgabe: eineZahl + " ist kleiner als " + a
+ Zeilenumbruch

Übung 2a: foreach-Schleife nach Struktogramm programmieren

LÖSUNG

+testen(a:int):void

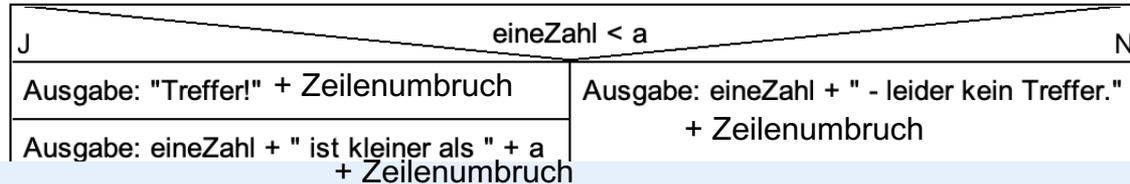
lokaleListe:ArrayList<Integer> = new ArrayList<Integer>()

17 zu lokaleListe hinzufügen

32 zu lokaleListe hinzufügen

11 zu lokaleListe hinzufügen

foreach(eineZahl:int : lokaleListe)



```
public void testen(int a) {  
    ArrayList<Integer> lokaleListe = new ArrayList<Integer>();  
    lokaleListe.add(17);  
    lokaleListe.add(32);  
    lokaleListe.add(11);  
    for(int eineZahl : lokaleListe) {  
        if(eineZahl < a) {  
            System.out.println("Treffer!");  
            System.out.println(eineZahl + " ist kleiner als " + a);  
        } else {  
            System.out.println(eineZahl + " - leider kein Treffer.");  
        }  
    }  
}
```

Übung 2b: Struktogramm anfertigen

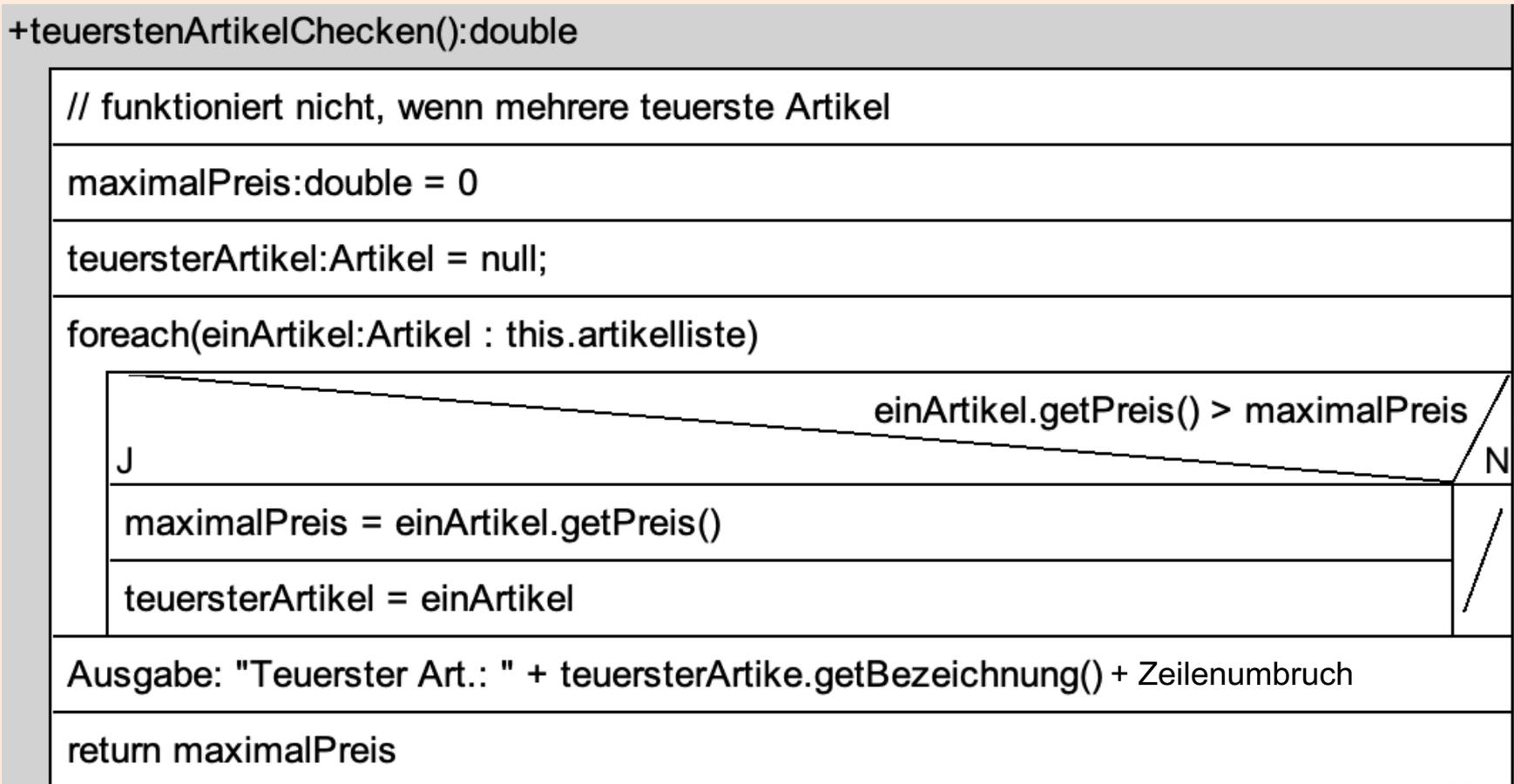
Fertigen Sie für die folgende Methode ein Struktogramm an:

```
public double teuerstenArtikelChecken() {  
    // Funktioniert nicht, wenn mehrere teuerste Artikel!  
    double maximalPreis = 0;  
    Artikel teuersterArtikel = null;  
    for(Artikel einArtikel : this.artikelliste) {  
        if(einArtikel.getPreis() > maximalPreis) {  
            maximalPreis = einArtikel.getPreis();  
            teuersterArtikel = einArtikel;  
        }  
    }  
    System.out.println("Teuerster Art.: " + teuersterArtikel.getBezeichnung());  
    return maximalPreis;  
}
```

Übung 2b: Struktogramm anfertigen

LÖSUNG

```
public double teuerstenArtikelChecken() {
    // Funktioniert nicht, wenn mehrere teuerste Artikel
    double maximalPreis = 0;
    Artikel teuersterArtikel = null;
    for(Artikel einArtikel : this.artikelliste) {
        if(einArtikel.getPreis() > maximalPreis) {
            maximalPreis = einArtikel.getPreis();
            teuersterArtikel = einArtikel;
        }
    }
    System.out.println("Teuerster Art.: " + teuersterArtikel.getBezeichnung());
    return maximalPreis;
}
```



Übung 6, PRO-VERSION:

Angenommen, es gibt mehrere teuerste Artikel, die alle den gleichen Preis haben. Erweitern Sie das Struktogramm so, dass eine Liste mit allen teuersten Artikeln zurückgegeben wird.

+teuerstenArtikelChecken():double

// funktioniert nicht, wenn mehrere teuerste Artikel

maximalPreis:double = 0

teuersterArtikel:Artikel = null;

foreach(einArtikel:Artikel : this.artikelliste)

einArtikel.getPreis() > maximalPreis

J

N

maximalPreis = einArtikel.getPreis()

teuersterArtikel = einArtikel

Ausgabe: "Teuerster Art.: " + teuersterArtike.getBezeichnung()

return maximalPreis

Übung 6, PRO-VERSION:

LÖSUNG

Angenommen, es gibt mehrere teuerste Artikel, die alle den gleichen Preis haben. Erweitern Sie das Struktogramm so, dass eine Liste mit allen teuersten Artikeln zurückgegeben wird.

```
+teuerstenArtikelChecken():ArrayList<Artikel>
```

```
maximalPreis:double = 0
```

```
// erst mal den Maximalpreis herausfinden
```

```
foreach(einArtikel:Artikel : this.artikelliste)
```

```
    einArtikel.getPreis() > maximalPreis
```

```
    J
```

```
    N
```

```
        maximalPreis = einArtikel.getPreis()
```

```
// jetzt alle Artikel, die den Maximalpreis haben, einer Liste hinzufügen
```

```
listeMitTeuerstenArtikeln:ArrayList<Artikel> = new ArrayList<Artikel>()
```

```
foreach(einArtikel:Artikel : this.artikelliste)
```

```
    einArtikel.getPreis() >= maximalPreis
```

```
    J
```

```
    N
```

```
        einArtikel der listeMitTeuerstenArtikeln hinzufügen
```

```
return listeMitTeuerstenArtikeln
```