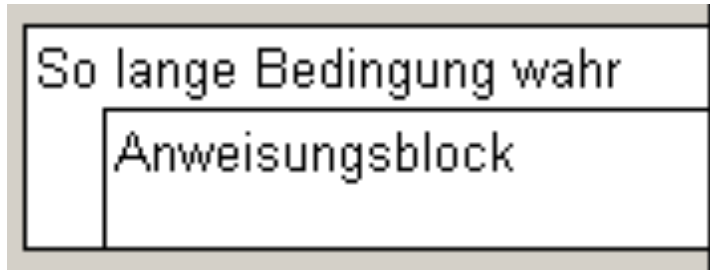


Java Schleifen:

while / do...while

kopfgesteuerte Schleife ("while ...")



*Bedingung wird im SchleifenKOPF geprüft
(d.h.: **vor** erstmaliger Ausführung d. Anweisungsblocks)*

fußgesteuerte Schleife ("do while")



*Bedingung wird im SchleifenFUSS geprüft
(d.h.: **nach** erstmaliger Ausführung d. Anweisungsblocks)
→ Fußgesteuerte Schleife wird immer mindestens 1x ausgeführt*

Syntax von while:

```
while (Bedingung)
{
    Anweisung
}
```

Beispiel: Zahlen von 0 bis 10 ausgeben

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

1
2
3
4
5
6
7
8
9
10

Syntax von while:

```
while (Bedingung)
{
    Anweisung
}
```

Beispiel: Zahlen von 0 bis 10 ausgeben

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

1
2
3
4
5
6
7
8
9
10

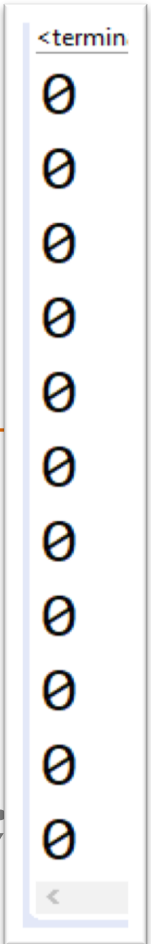
Abbruchbedingung

Syntax von while:

```
while (Bedingung)
{
    Anweisung
}
```

Beispiel: Zahlen von 0 bis 10 ausgeben

```
int zaehler = 0;
while (zaehler <= 10)
{ System.out.println(zaehler);
  // zaehler++; }
```



Häufigster Fehler:

Abbruchbedingung wird nicht erfüllt

→ Programm gerät in Endlosschleife

Syntax von do ... while:

```
do
```

```
{
```

```
    Anweisung
```

```
}
```

```
while (Bedingung)
```

```
// fortfahren, sobald Bedingung falsch ist!
```

Syntax von do ... while:

```
do
{
    Anweisung
}
while (Bedingung)
```

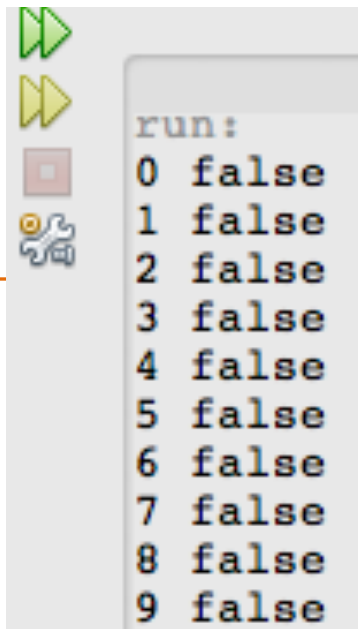
Beispiel: Zahlen von 0 bis 9 ausgeben

```
int zaehler = 0;
do
{
    System.out.println(zaehler);
    zaehler++;
}
while(zaehler < 10)
```

Beispiel: Zahlen von 0 bis 9, Abbruchbedingung mit boolscher Variable

```
int zaehler = 0;
boolean abbruchbedingung = false;

do
{
    System.out.print(zaehler);
    System.out.println(" " + abbruchbedingung);
    zaehler++;
    if (zaehler == 10)
    {
        abbruchbedingung = true;
    }
}
while (!abbruchbedingung);
// oder: while (abbruchbedingung == false);
```



run:

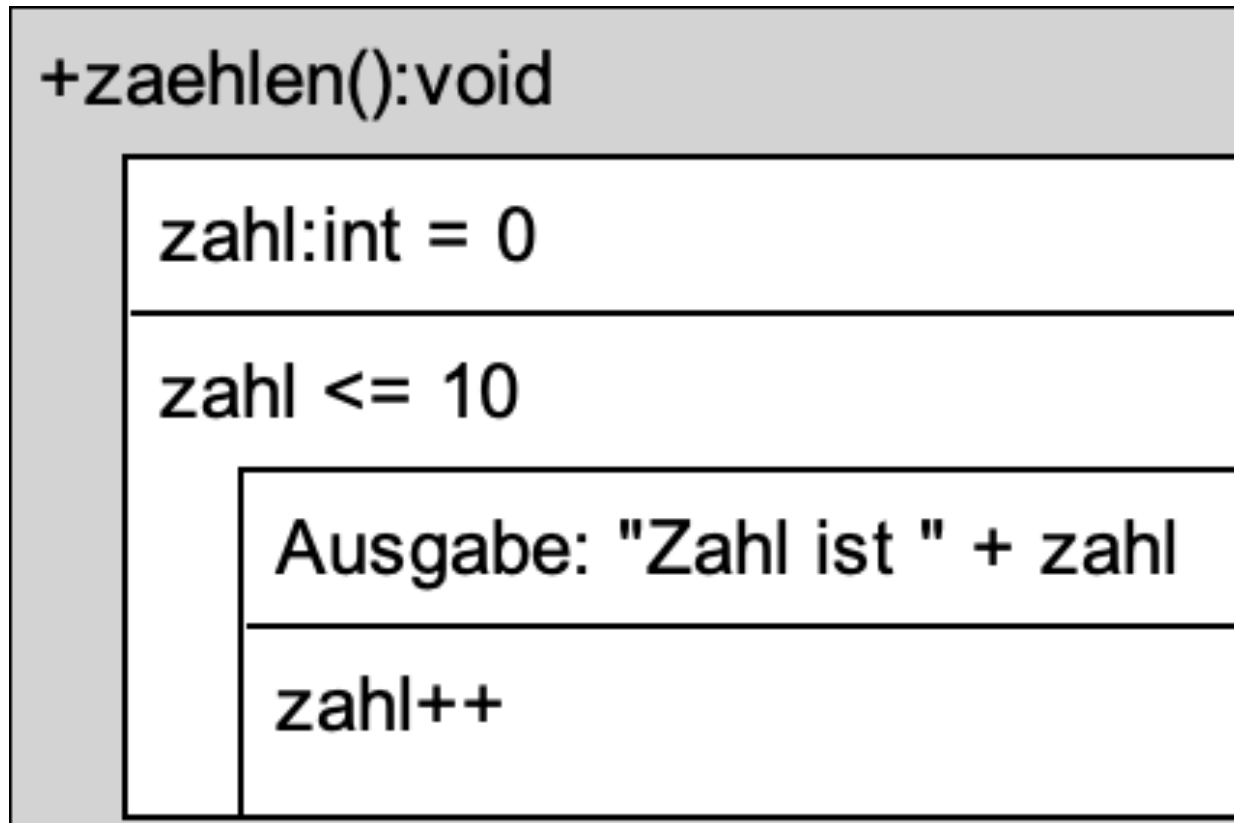
0	false
1	false
2	false
3	false
4	false
5	false
6	false
7	false
8	false
9	false

Bedingungen verknüpfen

```
// UND-Verknüpfung
while (x > 0 && y > 0)
{
    //
}
```

```
// ODER-Verknüpfung
while (x > 0 || y > 0)
{
    // ...
}
```

While-Schleife im Struktogramm



Übung - while-Schleife

Lassen Sie sich die Zahlen von 1 bis 55 untereinander ausgeben.

Danach kommt die Meldung „Fertig“

Erstellen Sie ein Struktogramm und setzen Sie es programmiertechnisch um.

Übung - while-Schleife - LÖSUNG

LÖSUNG

Lassen Sie sich die Zahlen von 1 bis 55 untereinander ausgeben.
Danach kommt die Meldung „Fertig“

```
+zahlenAusgeben():void
```

```
i:int = 1
```

```
i <= 55
```

```
Ausgabe: i + Zeilenumbruch
```

```
i++
```

```
Ausgabe: "Fertig"
```

```
public void zahlenAusgeben() {  
    int i = 1;  
    while (i <= 55) {  
        System.out.println(i);  
        i++;  
    }  
    System.out.println("Fertig");  
}
```

Wo liegt der Fehler beim folgenden Programm?

```
+zahlenAusgeben():void
```

```
i:int = 1
```

```
i <= 55
```

```
    Ausgabe: i
```

```
Ausgabe "Fertig"
```

Wo liegt der Fehler beim folgenden Programm?

LÖSUNG

```
+zahlenAusgeben():void
```

```
i:int = 1
```

```
i <= 55
```

```
    Ausgabe: i
```

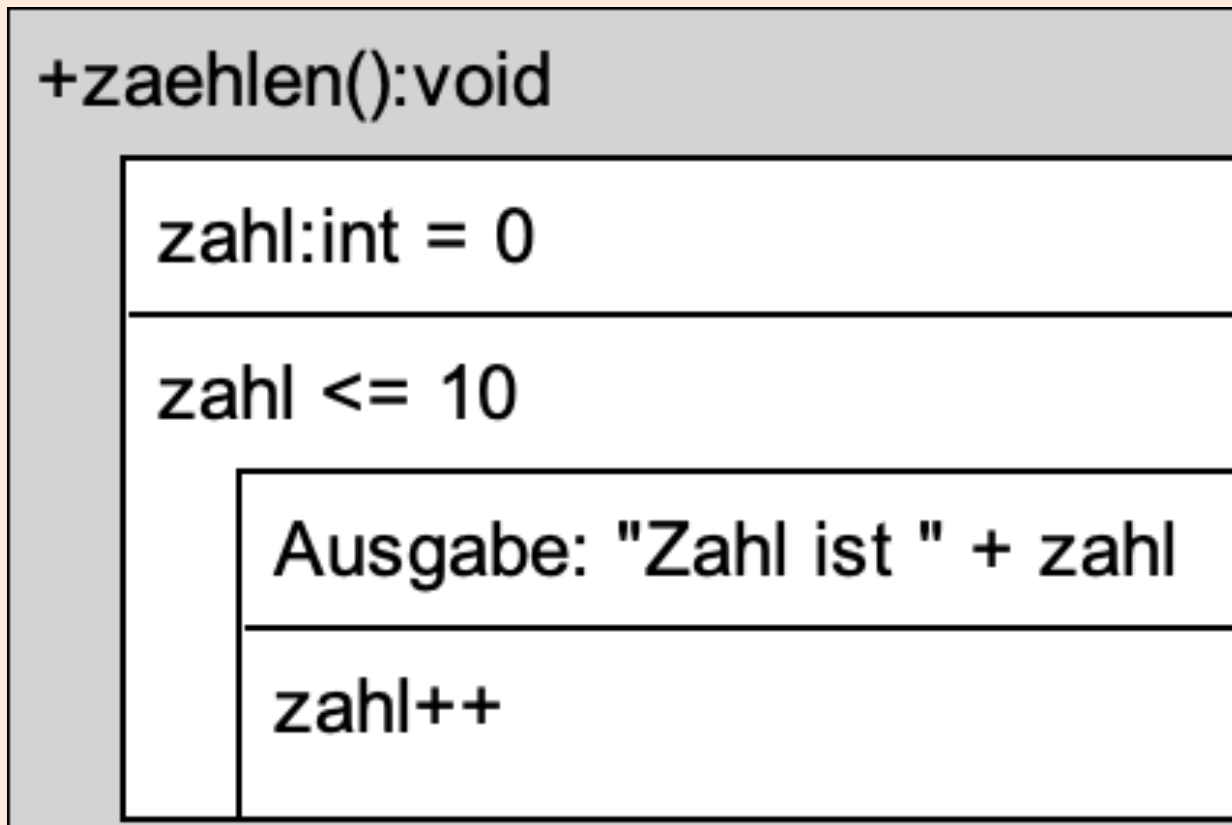
```
Ausgabe "Fertig"
```

i hat immer den gleichen Wert (nämlich 1), da i nicht verändert (z.B. hochgezählt) wird.

Die Schleife läuft also ewig, da die Abbruchbedingung ($i > 55$) niemals erfüllt wird.

Übung 1: Einfache while-Schleife programmieren

Setzen Sie das vorgegebene Struktogramm programmiertechnisch um und testen Sie die Funktionsweise.



Übung 1: Einfache while-Schleife programmieren

LÖSUNG

```
+zaehlen():void
```

```
zahl:int = 0
```

```
zahl <= 10
```

```
Ausgabe: "Zahl ist " + zahl
```

```
zahl++
```

```
public void zaehlen() {  
    int zahl = 0;  
    while(zahl <= 10) {  
        System.out.println("Zahl ist " + zahl);  
        zahl++;  
    }  
}
```


Übung 2: Einfache while-Schleife mit Parametern programmieren

Setzen Sie das vorgegebene Struktogramm programmiertechnisch um und testen Sie die Funktionsweise.

+zaehlenVonBis(start:int, ende:int, schrittweite:int):void

zaehler:int = start

J

start < ende

N

zaehler <= ende

Ausgabe: zaehler

zaehler = zaehler +
schrittweite

zaehler >= ende

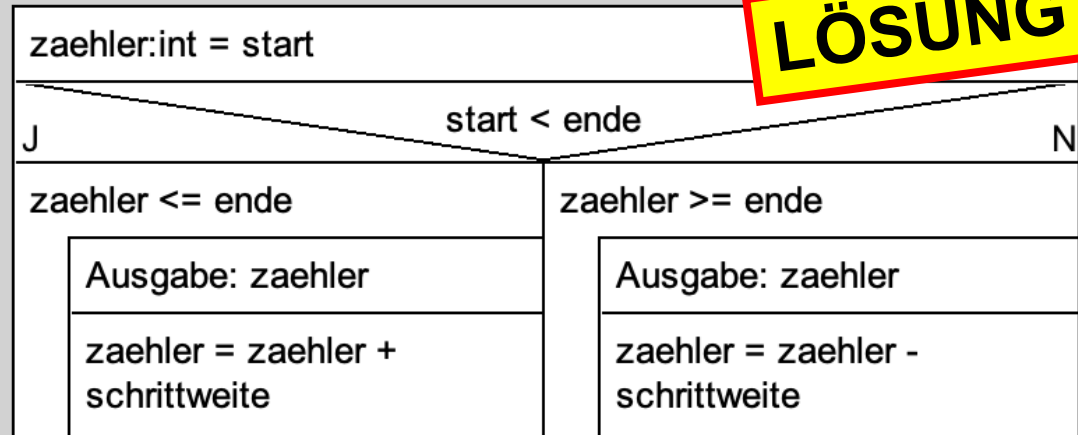
Ausgabe: zaehler

zaehler = zaehler -
schrittweite

Übung 2: Einfache while-Schleife mit Parametern programmieren

+zaehlenVonBis(start:int, ende:int, schrittweite:int):void

LÖSUNG



```
public void zaehlenVonBis(int start, int ende, int schrittweite) {
    int zaehler = start;
    if(start < ende) {
        while(zaehler <= ende) {
            System.out.println(zaehler);
            zaehler = zaehler + schrittweite;
        }
    } else {
        while(zaehler >= ende) {
            System.out.println(zaehler);
            zaehler = zaehler - schrittweite;
        }
    }
}
```

Übung 3: Programm in Struktogramm umsetzen

Setzen Sie das vorgegebene Programm in ein Struktogramm um.

```
public void whileSchleifen() {  
    String[] schuelerliste = {"Maria", "Marianne", "Marlene", "Mara"};  
    int index = 0;  
    while(index < schuelerliste.length) {  
        System.out.println(schuelerliste[index]);  
        index++;  
    }  
}
```

Übung 3: Programm in Struktogramm umsetzen

LÖSUNG

```
public void whileSchleifen() {  
    String[] schuelerliste = {"Maria", "Marianne", "Marlene", "Mara"};  
    int index = 0;  
    while(index < schuelerliste.length) {  
        System.out.println(schuelerliste[index]);  
        index++;  
    }  
}
```

+whileSchleifen():void

schuelerliste:String[] = {"Maria", "Marianne", "Marlene", "Mara"}

index:int = 0

index < schuelerliste.length

Ausgabe: schuelerliste[index]

index++

Übung 4: Programm in Struktogramm umsetzen

Setzen Sie das vorgegebene Programm in ein Struktogramm um.

```
public void arraylistDurchlaufen() {  
    ArrayList<String> namensliste = new ArrayList<String>();  
    namensliste.add("Sandy");  
    namensliste.add("Chantal");  
    namensliste.add("Alfons");  
    int laufvariable = 0;  
    while(laufvariable < namensliste.size()) {  
        System.out.println(namensliste.get(laufvariable));  
        laufvariable++;  
    }  
}
```

Übung 4: Programm in Struktogramm umsetzen

Setzen Sie das vorgegebene Programm in ein Struktogramm um.

LÖSUNG

```
public void arraylistDurchlaufen() {  
    ArrayList<String> namensliste = new ArrayList<String>();  
    namensliste.add("Sandy");  
    namensliste.add("Chantal");  
    namensliste.add("Alfons");  
    int laufvariable = 0;  
    while(laufvariable < namensliste.size()) {  
        System.out.println(namensliste.get(laufvariable));  
        laufvariable++;  
    }  
}
```

+arraylistDurchlaufen():void

namensliste:ArrayList<String> = new ArrayList<String>()

"Sandy" der namensliste hinzufügen

"Chantal" der namensliste hinzufügen

"Alfons" der namensliste hinzufügen

int:laufvariable = 0

laufvariable < Größe von namensliste

Ausgabe: namensliste.get(laufvariable) + Zeilenumbruch

laufvariable++