

4. Schleifen

4.1 While-Schleife

4.2 Do-While-Schleife

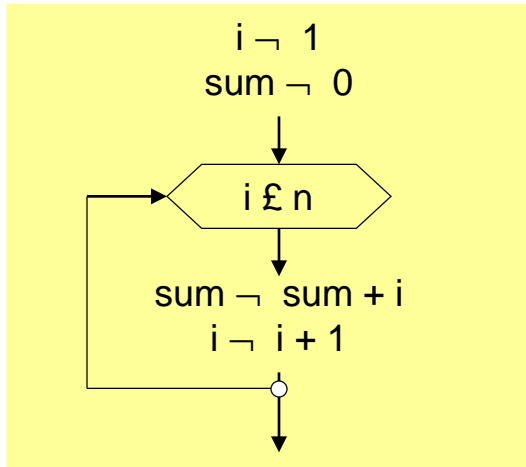
4.3 For-Schleife

4.4 Abbruch von Schleifen

4.5 Vergleich der Schleifenarten

While-Schleife

Führt eine Anweisungsfolge aus, solange eine bestimmte Bedingung gilt



```

i = 1;
sum = 0;
while (i <= n) {
    sum = sum + i;
    i = i + 1;
}
  
```

Schleifenkopf
mit Schleifenbedingung

Schleifenrumpf

Schreibtischtest

n	i	sum
3	1	0
	2	1
	3	3
	4	6

Syntax

Statement = Assignment | IfStatement | SwitchStatement | WhileStatement | ... | Block.

WhileStatement = **"while"** "(" BoolExpr ")" Statement .

Wenn Schleifenrumpf aus mehreren Anweisungen besteht, muss er mit {...} geklammert werden.

Beispiel



Aufgabe: Zahlenfolge lesen und Histogramm ausgeben

Eingabe: 3 2 5

Ausgabe: ***
**

```
class Histogram {  
  
    public static void main (String[] arg) {  
        int i = In.readInt();  
        while (In.done()) {  
            int j = 1;  
            while (j <= i ) {Out.print("*"); j++;}  
            Out.println();  
            i = In.readInt();  
        }  
    }  
}
```

liest die Zahlenfolge

gibt i Sterne aus

Schreibtischsimulation

i	j	
3	1	*
	2	*
	3	*
2	1	*
	2	*
	3	*
...		

Assertionen bei Schleifen

Triviale Assertionen

Aussagen, die sich aus der Schleifenbedingung ergeben

```
i = 1; sum = 0;
while (i <= n) { /* i <= n */
    sum = sum + i;
    i = i + 1;
}
/* i > n */
```

sollte man immer hinschreiben
oder zumindest im Kopf bilden

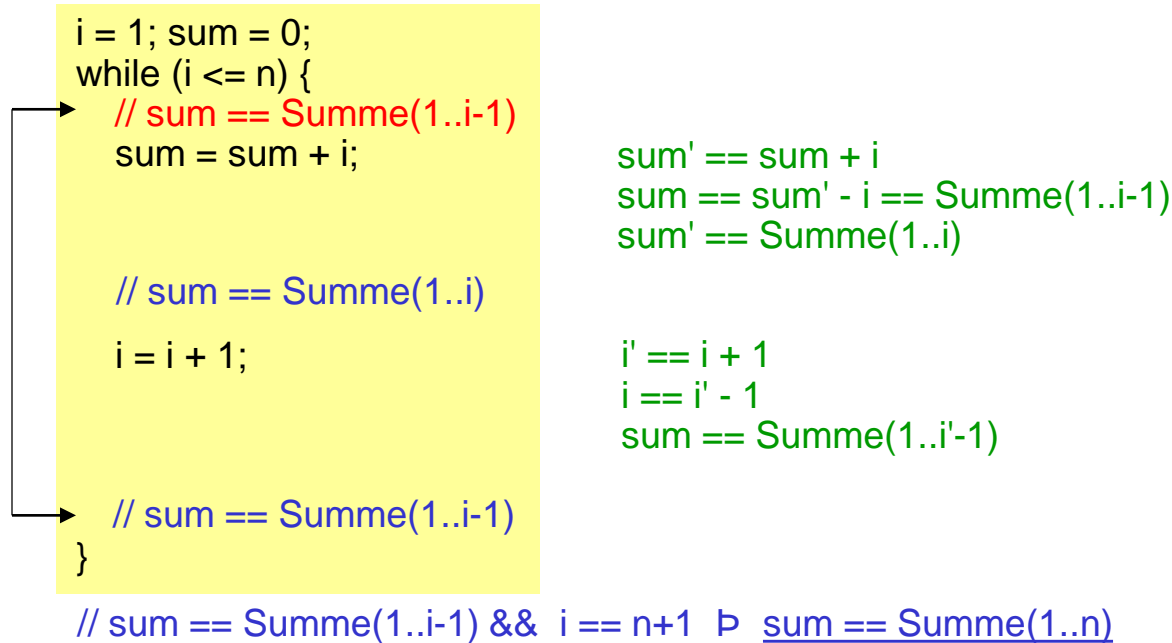
Schleifeninvariante

Aussage über das berechnete Ergebnis, die in jedem Schleifendurchlauf gleich bleibt

```
i = 1; sum = 0;
while (i <= n) { /* i <= n */
    /* sum == Summe(1..i-1) */
    sum = sum + i;
    i = i + 1;
}
/* i > n */
```

Verifikation der Schleife

"Durchdrücken" der Invariante durch die Anweisungen des Schleifenrumpfs



Termination der Schleife muss auch noch bewiesen werden:

i wird in jedem Durchlauf erhöht und ist mit n beschränkt

4. Schleifen

4.1 While-Schleife

4.2 Do-While-Schleife

4.3 For-Schleife

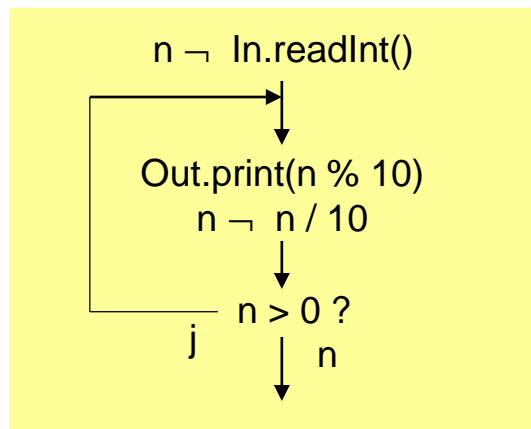
4.4 Abbruch von Schleifen

4.5 Vergleich der Schleifenarten

Do-While-Schleife

Schleifenbedingung wird *am Ende der Schleife* geprüft

Beispiel: Ausgabe der Ziffern einer Zahl in umgekehrter Reihenfolge



```

int n = In.readInt();
do {
    Out.print(n % 10);
    n = n / 10;
} while ( n > 0 );
  
```

Schreibtischtest

n	n % 10
123	3
12	2
1	1
0	

Syntax

Statement = Assignment | IfStatement | WhileStatement | **DoWhileStatement** | ... | Block.

DoWhileStatement = "**do**" Statement "**while**" "(" BoolExpr ")" ";".

Wenn Schleifenrumpf aus mehreren Anweisungen besteht, muss er mit {...} geklammert werden.

Do-While-Schleife



Warum kann man dieses Beispiel nicht mit einer While-Schleife lösen?

```
int n = In.readInt();  
while (n > 0) {  
    Out.print(n % 10);  
    n = n / 10;  
}
```

"Abweisschleife"

Weil das für $n == 0$ die falsche Ausgabe liefern würde.
Die Schleife muss mindestens einmal durchlaufen werden, daher:

```
int n = In.readInt();  
do {  
    Out.print(n % 10);  
    n = n / 10;  
} while (n > 0);
```

"Durchlaufschleife"

Meist ist eine While-Schleife das passende Konstrukt.

4. Schleifen

- 4.1 While-Schleife
- 4.2 Do-While-Schleife
- 4.3 For-Schleife
- 4.4 Abbruch von Schleifen
- 4.5 Vergleich der Schleifenarten

For-Schleife (Zählschleife)

Falls die Anzahl der Schleifendurchläufe im voraus bekannt ist

```
sum = 0;
for ( i = 1 ; i <= n ; i++ )
    sum = sum + i;
```

- 1) Initialisierung der Laufvariablen
- 2) Schleifenabbruchbedingung
- 3) Ändern der Laufvariablen

Ist eine Kurzform für

```
sum = 0;
i = 1;
while ( i <= n ) {
    sum = sum + i;
    i++;
}
```

Syntax der For-Schleife



```
ForStatement = "for" "(" [ForInit] ";" [BoolExpr] ";" [ForUpdate] ")" Statement.  
ForInit      = Assignment {"", " Assignment}  
              | Type VarDecl {"", " VarDecl}.  
ForUpdate    = Assignment {"", " Assignment}.
```

Beispiele

```
for (i = 0; i < n; i++) ...
```

// i: 0, 1, 2, 3, ..., n-1

```
for (i = 10; i > 0; i--) ...
```

// i: 10, 9, 8, 7, ..., 1

```
for (int i = 0; i <= n; i = i + 1) ...
```

// i: 0, 1, 2, 3, ..., n

```
for (int i = 0, j = 0; i < n && j < m; i = i + 1, j = j + 2) ...
```

// i: 0, 1, 2, 3, ... } terminiert, sobald
// j: 0, 2, 4, 6, ... } i >= n || j >= m

```
for (;;) ...
```

// Endlosschleife

Beispiel: Multiplikationstabelle drucken



```
class PrintMulTab {  
  
    public static void main (String[] arg) {  
        int n = In.readInt();  
        for (int i = 1; i <= n; i++) {  
            for (int j = 1; j <= n; j++) {  
                Out.print(i * j + " ");  
            }  
            Out.println();  
        }  
    }  
}
```

1	2	3
2	4	6
3	6	9

Schreibtischtest für n == 3

i	j	
1	1	1 2 3
	2	2 4 6
	3	3 6 9
	4	
2	1	
	2	
	3	
	4	
3	1	
	2	
	3	
	4	
4	1	
	2	
	3	
	4	

4. Schleifen

- 4.1 While-Schleife
- 4.2 Do-While-Schleife
- 4.3 For-Schleife
- 4.4 Abbruch von Schleifen
- 4.5 Vergleich der Schleifenarten

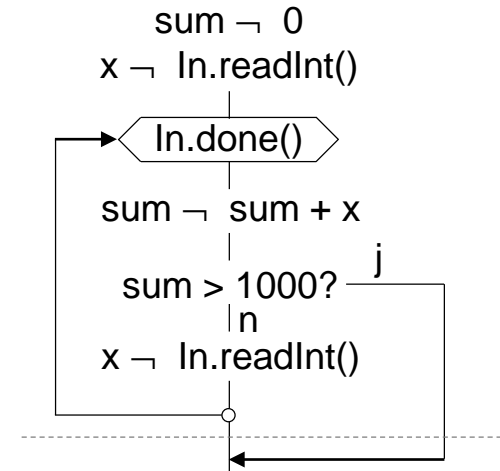
Abbruch von Schleifen



Beispiel: Summieren mit Fehlerabbruch

```
int sum = 0;
int x = In.readInt();
while (In.done()) {
    sum = sum + x;
    if (sum > 1000) { Out.println("zu gross"); break; }
    x = In.readInt();
}
```

break verlässt die Schleife,
in der es enthalten ist
(*while, do, for*)



Baustein mit 2 Ausgängen!

Schleifenabbruch mit *break* möglichst vermeiden: schwer zu verifizieren.

Meist lässt sich dasselbe auch mit *while* ausdrücken:

```
int sum = 0;
int x = In.readInt();
while (In.done() && sum <= 1000) {
    sum = sum + x;
    if (sum > 1000) Out.println("zu gross"); else x = In.readInt();
}
// ! In.done() || sum > 1000
```

Abbruch äußerer Schleifen



Beispiel

```
outer:                                // Marke!  
for (;;) {                            // Endlosschleife!  
    for (;;) {  
        ...  
        if (...) break;              // verlässt innere Schleife  
        else if (...) break outer;  // verlässt äußere Schleife  
        ...  
    }  
}
```

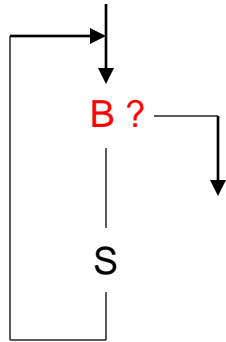
Wann ist ein Schleifenabbruch mit break vertretbar?

- bei Abbruch wegen Fehlern
- bei mehreren Aussprüngen an verschiedenen Stellen der Schleife
- bei echten Endlosschleifen (z.B. in Echtzeitsystemen)

4. Schleifen

- 4.1 While-Schleife
- 4.2 Do-While-Schleife
- 4.3 For-Schleife
- 4.4 Abbruch von Schleifen
- 4.5 Vergleich der Schleifenarten

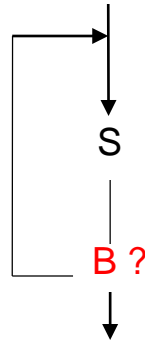
Vergleich der Schleifenarten



Abweisschleife

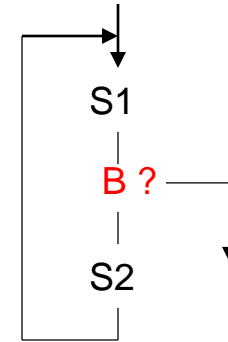
```
while (B)  
  S
```

```
for (I; B; U)  
  S
```



Durchlaufschleife

```
do  
  S  
while (B)
```



allgemeine Schleife

```
for (;;) {  
  S1;  
  if (B) break;  
  S2;  
}
```