

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging



Folien: go.tum.de/904005

MiniJava

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

Deklaration von Variablen

```
int x, y;
```

Zuweisung von Werten

```
x = 5; //Variable x wird Wert 5 zugewiesen
```

```
y = x + 2; //y wird x + 2 (7) zugewiesen
```

```
x = x + 1; //x wird um 1 inkrementiert (6)
```

MiniJava

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

Einlesen von Werten über die Konsole

```
int x;  
x = readInt();  
x = readInt("text");
```

Ausgabe von Werten auf die Konsole

```
write(x);  
write("text");  
writeConsole(x); //ohne Zeilenumbruch
```

MiniJava

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

Selektion

```
int x, y;
```

```
x = readInt(); //Einlesen von x und y
```

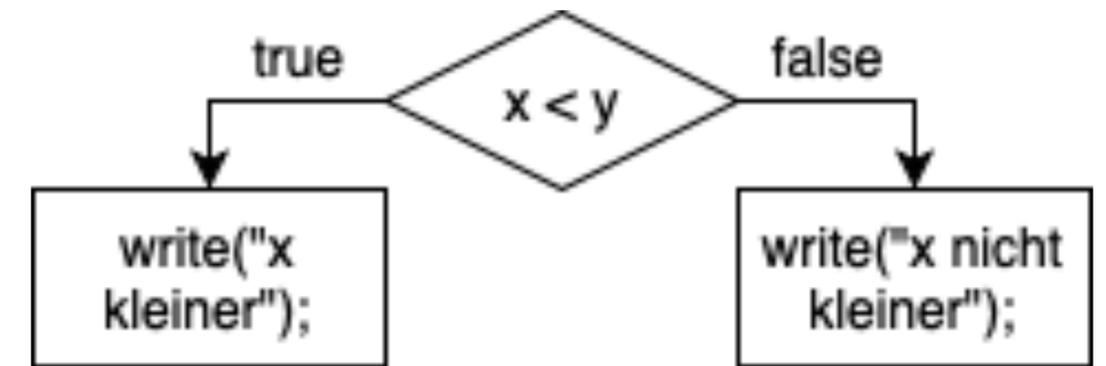
```
y = readInt();
```

```
if (x < y) //Überprüfen der Bed.
```

```
    write("x ist kleiner");
```

```
else
```

```
    write("x ist nicht kleiner");
```



MiniJava

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

Selektion

```
int x, y;  
x = readInt(); //Einlesen von x und y  
y = readInt();  
if (x < y) { //Überprüfen der Bed.  
    write("x ist kleiner");  
}  
else {  
    write("x ist nicht kleiner");  
}
```

Nur nötig, falls mehr
als eine Operation
auf if/else folgt.

MiniJava

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

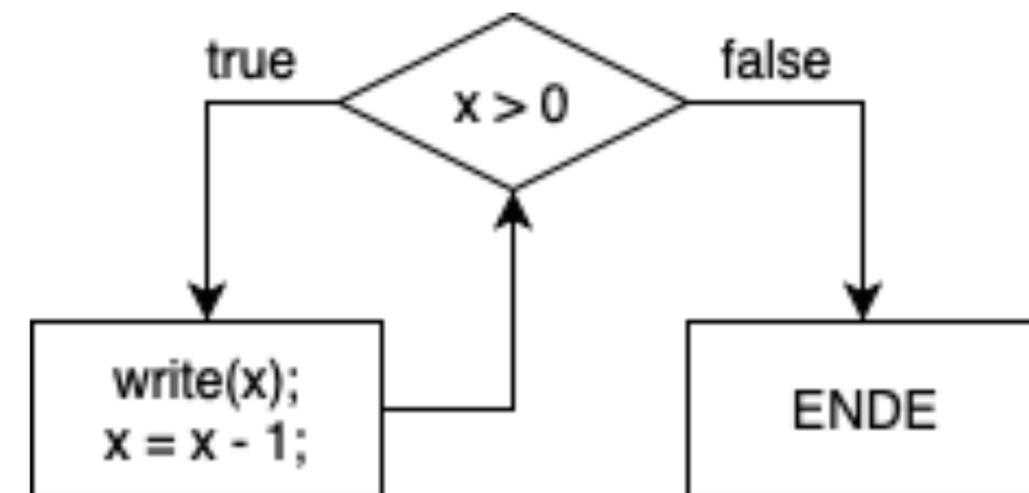
Debugging

Iteration

```
int x;
```

```
x = readInt();
```

```
while (x > 0) { //Überprüfen der Bedingung  
    write(x); //Ausgabe von x  
    x = x - 1; //Dekrementieren von x  
}
```



MiniJava

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

Arrays

```
int [] a = new int[4]; //Array d. Länge 4 dekl.  
a[0] = 2; //Zuweisung an 1. Stelle des Arrays  
a[1] = 5; a[2] = 1;  
a[3] = a[0] + a[1] + a[2];
```

Index	0	1	2	3
Wert	2	3	1	6

Arrays werden
von 0 bis length – 1
indiziert.

Aufbau von Klassen

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

```
public class Test extends MiniJava {  
    public static void main(String[] args) {  
        // "Hauptprogramm"  
  
    }  
    // weitere Methoden  
}
```

Aufbau von Klassen

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

```
public class Test extends MiniJava {  
    public static void main(String[] args) {  
        // "Hauptprogramm"  
  
    }  
    // weitere Methoden  
}
```

Vererbung: So bekommt man die speziellen MiniJava Methoden. (später mehr dazu)

Kommentare

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

Wir kommentieren den Code, damit das Programm für uns & andere übersichtlicher und leichter zu lesen ist.

Einzeiliger Kommentar

```
//Kommentar
```

Mehrzeiliger Kommentar

```
/* Kommentar über  
mehrere Zeilen */
```

Syntax

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

Beliebte Klausuraufgaben

- Syntaxbäume
- Programme auf Korrektheit prüfen

[Übersicht \(von SS2011\)](#)

Kontrollfluss

MiniJava

Aufbau von Klassen

Kommentare

Syntax

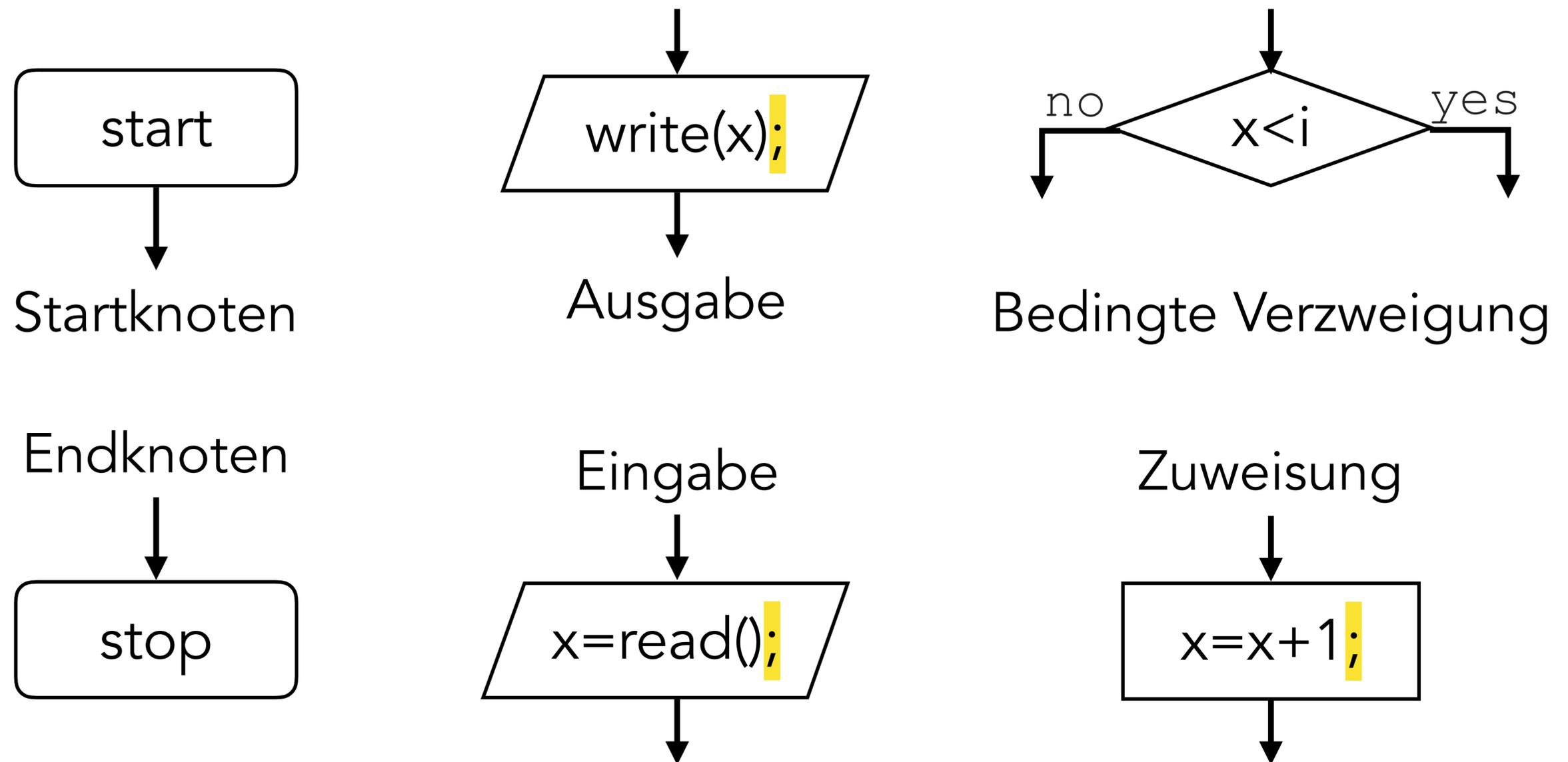
Kontrollfluss

Basistypen

P-Aufgaben

Debugging

Beliebte Klausuraufgabe: Kontrollflussdiagramme



Basistypen

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

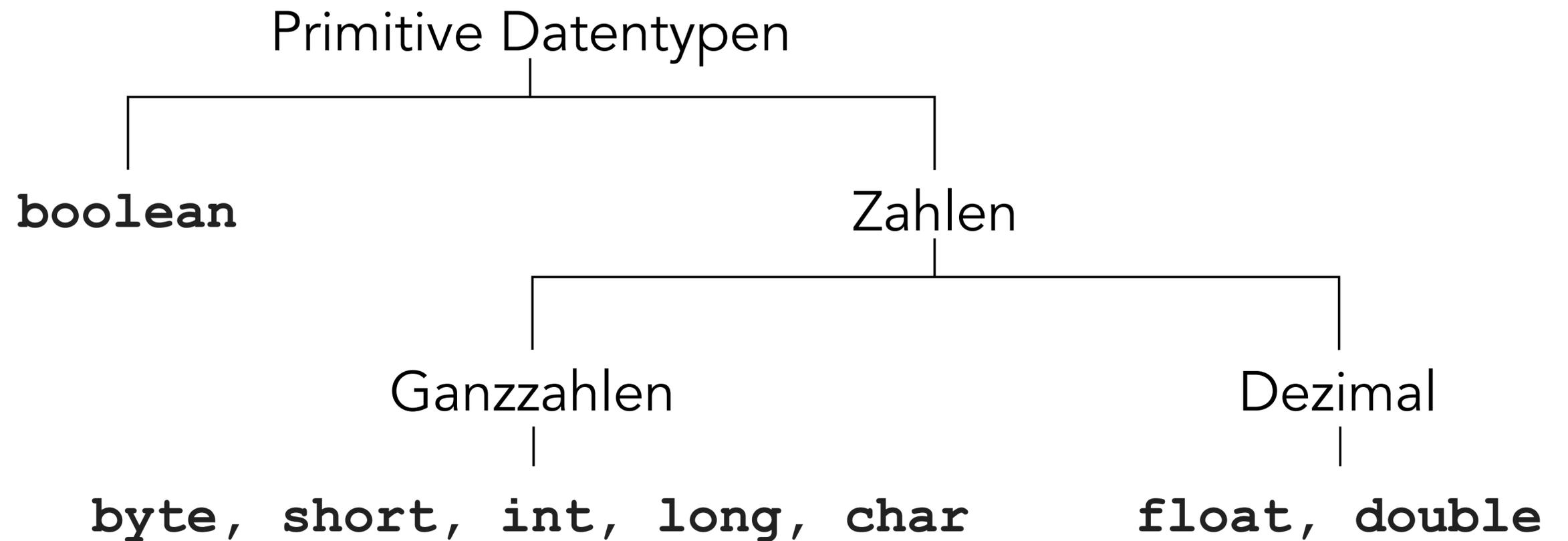
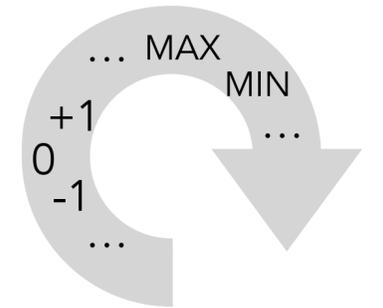
Basistypen

P-Aufgaben

Debugging

Achtung! Überlauf von Integer:

`Integer.MAX_VALUE + 1 = Integer.MIN_VALUE`



P02.1

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

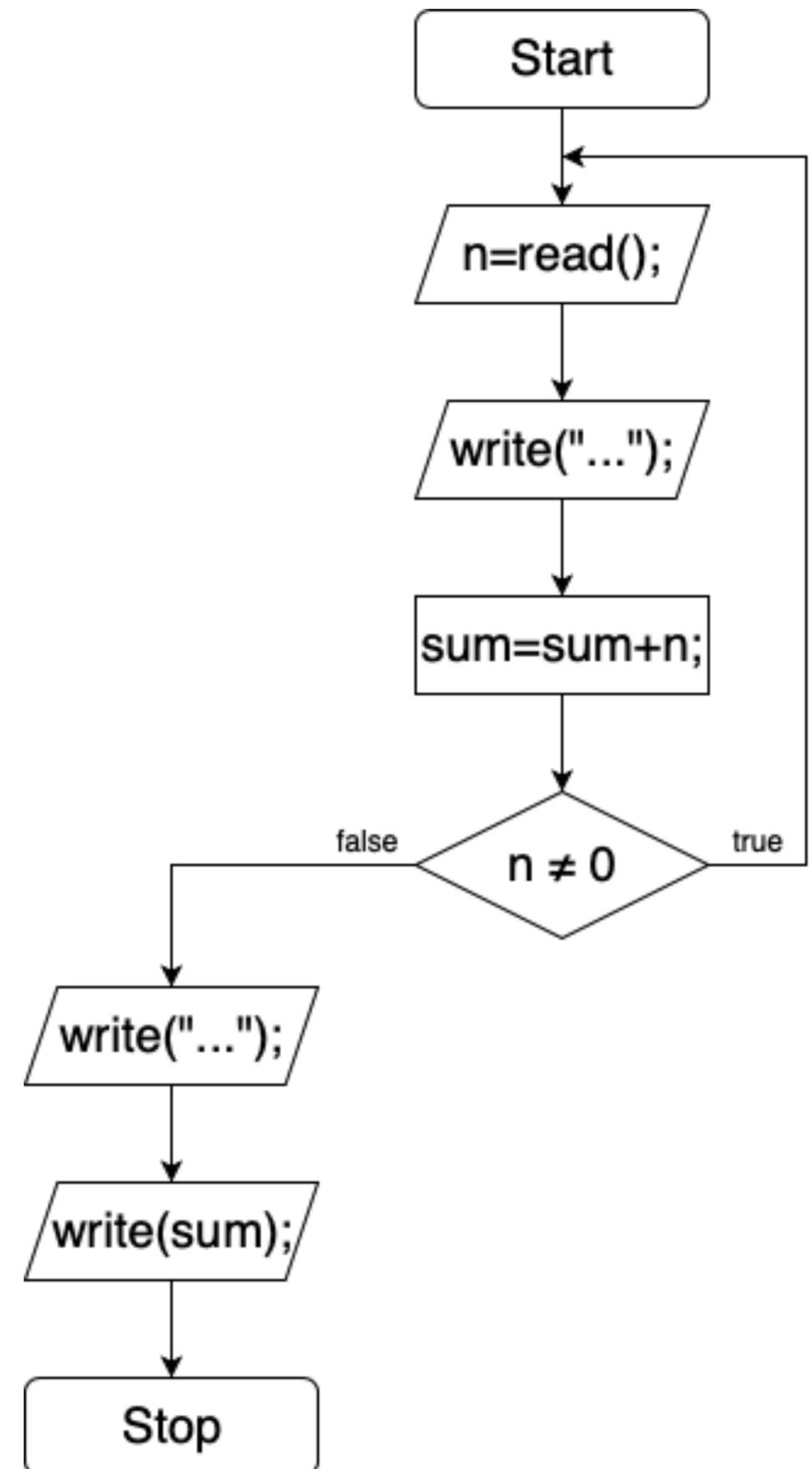
P-Aufgaben

Debugging

Do-While Schleife

```
do {  
    //Code  
} while (cond) ;
```

Die erste Iteration erfolgt ohne Prüfung der Bedingung.



P02.2

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

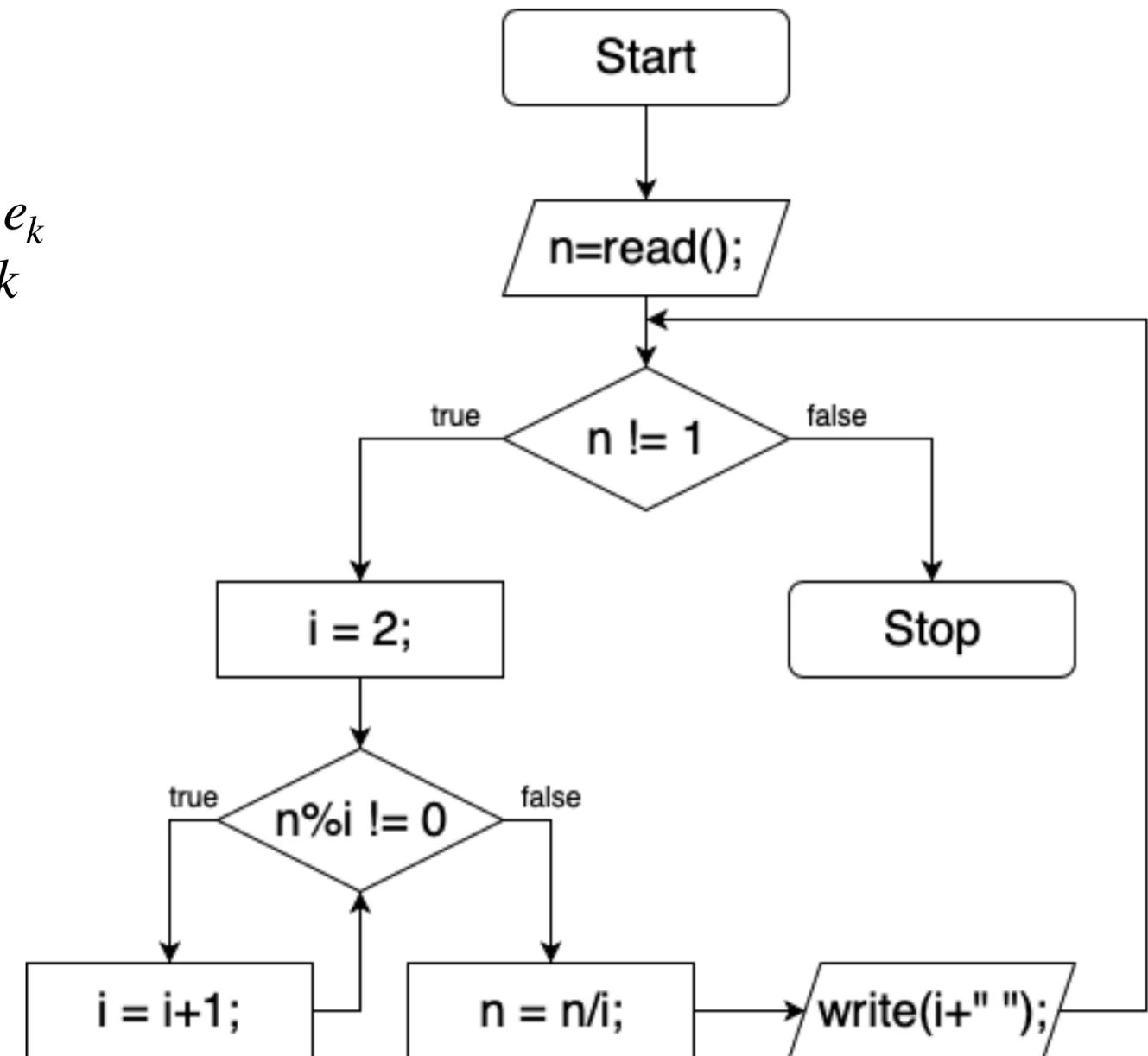
Basistypen

P-Aufgaben

Debugging

Primfaktorzerlegung: Darstellung von $n \in \mathbb{N}$ als Produkt von Primzahlen (\mathbb{P}).

$$n = p_1^{e_1} \cdot \dots \cdot p_M^{e_M} = \prod_{k=1}^M p_k^{e_k}$$



P02.3

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

Karnickelpopulation

- 3 Generationen
- Jede Generation bringt in jedem Zyklus (Monat) pro Paar ein neues Paar in *Generation 1*

Monat	1. Gen	2. Gen	3. Gen
1	1	0	0
2	1	1	0
3	2	1	1
4	4	2	1
5	7	4	2

T_n	T_{n+1}	Gen
j	$j+a+m$	J
m	j	M
a	m	A

P01.4

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

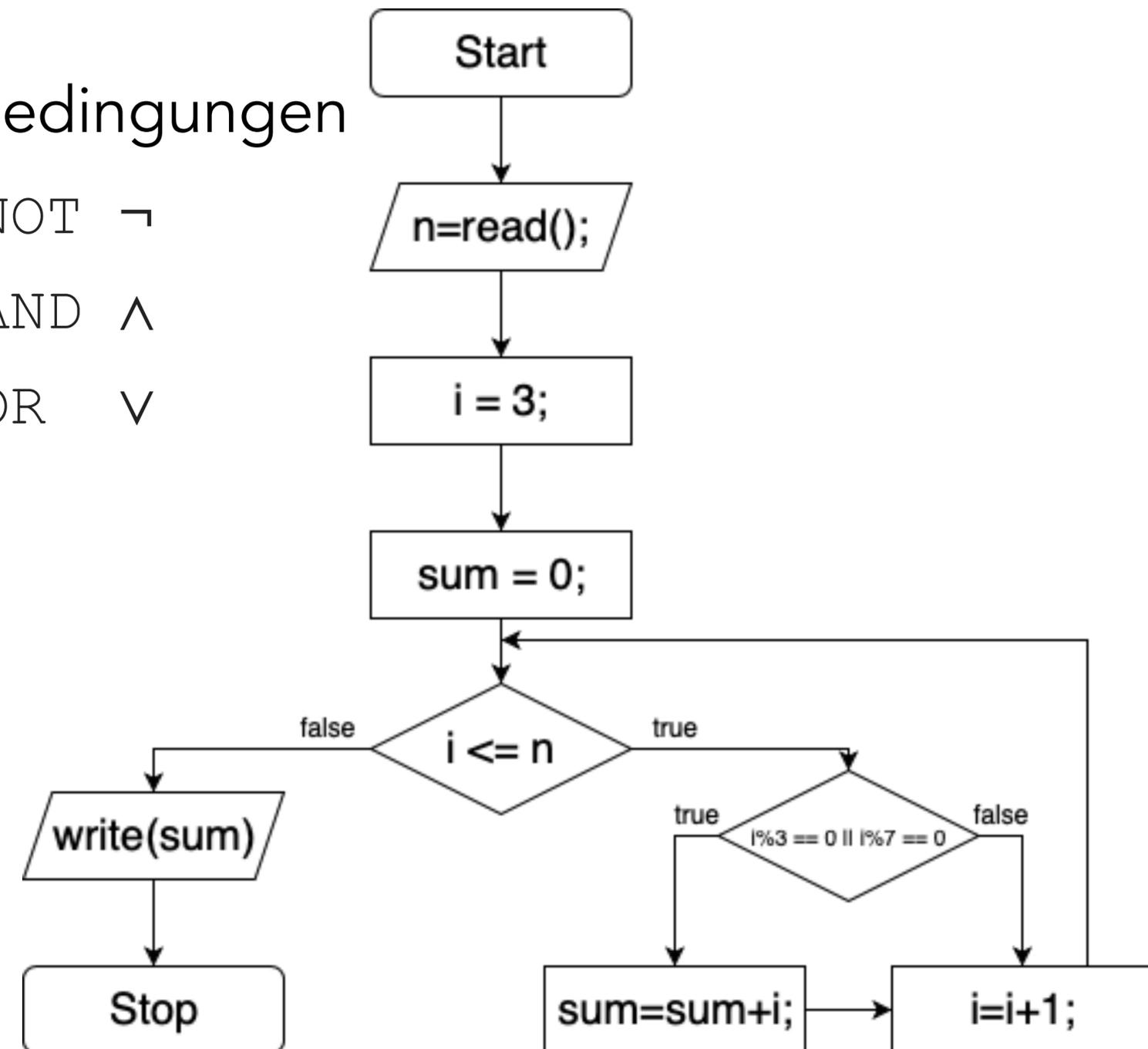
Debugging

NOT, AND und OR bei Bedingungen

`!a //logisches NOT \neg` `a && b //logisches AND \wedge` `a || b //logisches OR \vee`

&&	1	0
1	1	0
0	0	0

 	1	0
1	1	1
0	1	0



Debugging

MiniJava

Aufbau von Klassen

Kommentare

Syntax

Kontrollfluss

Basistypen

P-Aufgaben

Debugging

2. Debugmodus starten

3a. Durch Programm gehen...

1: Breakpoint setzen

The screenshot shows the Eclipse IDE with the following components:

- Editor:** Kanickel.java with the following code:


```

1 public class Kanickel extends MiniJava {
2
3
4     public static void main(String[] args) {
5
6         int jung = 0;
7         int mittel = 0;
8         int alt = 0;
9
10        jung = readInt("Bitte Zahl eingeben:");
11        for(int i = 0; i < 12; i++) {
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
      
```
- Debugger:** The program is paused at line 11. The console shows the prompt "Bitte Zahl eingeben:" and the input "5".
- Variables View:** Shows the following variables and values:

Name	Value
readInt() returned	5
args	String[0] (id=19)
jung	5
mittel	0
alt	0

3b. ...und dabei die Variablen beobachten