Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Programmiervorkurs
Wintersemester 20/21

Tag 2

Java

Agenda

- Workshops zur Wiederholung
- Tutorium
  - Scanner
  - Bedingte Anweisung (If/Else)
  - Vergleichsoperatoren
  - Logische Operatoren
  - Fallunterscheidungen (Switch)
  - Gültigkeit von Variablen
- Recherche
- Mäxchen 2.0 & Roboter & Aufgaben

Ergebnisse des Workshops

- Tipps für verständlichen Code
  - gut strukturierter Code erleichtert das Verständnis und verringert Fehler
    → IntelliJ-Shortcut für Auto-Formatierung: Strg + Umschalt + Alt + L
  - sinnvolle Variablen-Namen erleichtern das Verstehen des Programmcodes
- Modulo-Operator x % y liefert den Rest bei einer schriftlichen Division von x durch y
- Inkrement-Operatoren
  - ++ erhöht die Variable um eins (x++)
  - -- verringert die Variable um eins (x--)
  - Operator nach Variable: Ausführung nach Verwendung
  - Operator vor Variable: Ausführung vor Verwendung

SVN und Pabs

Windows

› SlikSVN

Mac

› Homebrew & SVN

Linux

› Paketmananger apt / dnf -> SVN

Anleitung:

› Wendet euch an das Tutorials “pabs.pdf“ im Kurs

› Im Unterpunkt 2.5

Benutzereingaben mit Scanner

› Bereits gelernt: Ausgabe von Text

```
System.out.println("Hallo Welt!");
```

› Neu: Einlesen von Text (Benutzereingabe)

› Benutzer kann Programmablauf selbst steuern

› Benutzer kann selbst Werte eingeben

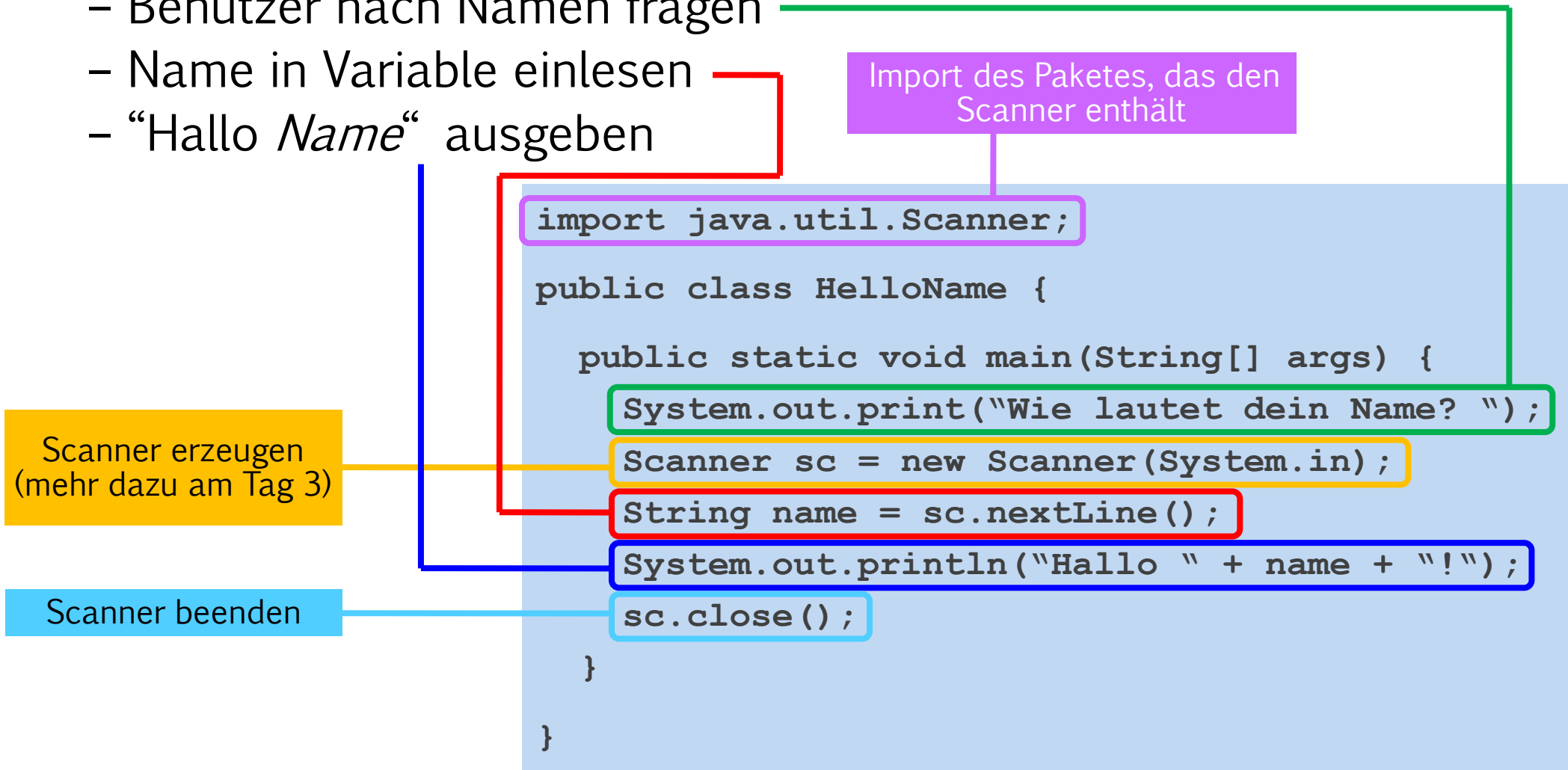
› Zum Einlesen: Objekt der Klasse **Scanner** erforderlich

Objektorientierung: Tag 3

Hallo Name

Programmablauf:

- Benutzer nach Namen fragen
- Name in Variable einlesen
- “Hallo *Name*“ ausgeben



Weitere Methoden zum Einlesen

Methode	Beschreibung	Einzulesender Datentyp
next()	Liest Eingabe bis zum ersten Leerzeichen	String
nextLine()	Liest ganze Zeile ein	String
nextInt()	Liest nächste ganze Zahl ein	Integer
nextDouble()	Liest nächste Dezimalzahl ein	Double
…	… "	…

Quelle: http://docs.oracle.com/javase/8/docs/api/java/util/Scanner.html

Hinweis: Wenn ein String nach einer Zahl eingelesen werden soll, muss stets ein nextLine() folgen

Hinweis: Je nach Systemeinstellung ist eventuell ein Punkt (.) als Dezimaltrennzeichen zu verwenden

Beispiel:

```
System.out.print("Gib eine Dezimalzahl ein: ");
Scanner sc = new Scanner(System.in);
double eingabe = sc.nextDouble();
sc.nextLine();
System.out.println("Du hast " + eingabe + " eingegeben.");
```

Aufgabe: Visitenkarte

GRUNDLAGE

› Erweitert das Programm von Tag 1 um die Personendaten für die Visitenkarte zuerst vom Benutzer abzufragen und anschließend auszugeben.

```
---------------------------------
Name:          Monika Mustermann
Straße:        Musterstraße 1a
Ort:           12345 Musterstadt
Telefon:       0123 456789
Geb.-Datum:    01.01.1970
Alter:         45
---------------------------------
```

ZUSATZ

› Schreibt ein kleines Programm, dass zwei Kommazahlen i und j und einen boolean b einließt.

› Im Anschluss wird der boolean und die Summe der Zahlen ausgegeben.

Bedingte Anweisung (If)

Ausführung eines Codeblocks nur, wenn eine Bedingung erfüllt ist

Beispiel:

```
System.out.print("Bitte geben Sie eine Augenzahl zwischen 1 und 6 ein: ");
Scanner sc = new Scanner(System.in);
int pips = sc.nextInt();
if (pips < 7) {
  System.out.println("Prima!");
}
sc.close();
```

Begrenzung des Codeblocks

Bedingte Anweisung (If, Else)

Ergänzung um eine sonstige Anweisung, die ausgeführt wird, wenn die Bedingung nicht zutrifft

Beispiel:

```
System.out.print("Bitte geben Sie eine Augenzahl zwischen 1 und 6 ein: ");
Scanner sc = new Scanner(System.in);
int pips = sc.nextInt();
if (pips < 7) {
  System.out.println("Prima!");
} else {
  System.out.println("Sie haben Blödsinn eingegeben!");
}
sc.close();
```

Bedingte Anweisung (If, Else If, Else)

Ergänzung um eine weitere Bedingung

Beispiel:

```
System.out.print("Bitte geben Sie eine Augenzahl zwischen 1 und 6 ein: ");
Scanner sc = new Scanner(System.in);
int pips = sc.nextInt();
if (pips < 1) {
  System.out.println("Die Augenzahl ist kleiner 1!");
} else if (pips > 6) {
  System.out.println("Die Augenzahl ist größer 6!");
} else {
  System.out.println("Prima!");
}
sc.close();
```

Vergleichsoperatoren

Operator	Beispiel	Math.	Ergibt für zwei Werte genau dann "true", wenn ...
`>`	`a > b`	$>$	... `a` größer als `b` ist
`<`	`a < b`	$<$	... `a` kleiner als `b` ist
`>=`	`a >= b`	$\geq$	... `a` größer als `b` oder `a` gleich `b` ist
`<=`	`a <= b`	$\leq$	... `a` kleiner als `b` oder `a` gleich `b` ist
`==`	`a == b`	$=$	... `a` gleich `b` ist
`!=`	`a != b`	$\neq$	... `a` ungleich `b` ist

Beispiel ≠

```
int geheimeZahl = 7;
System.out.print("Rate die geheime Zahl: ");
Scanner sc = new Scanner(System.in);
int gerateneZahl = sc.nextInt();
if (gerateneZahl != geheimeZahl) {
  System.out.println("Du hast falsch geraten!");
} else {
  System.out.println("Du hast richtig geraten!");
}
sc.close();
```

Vergleich von Strings

Falsch!

```
Scanner sc = new Scanner(System.in);
System.out.print("Bitte 1. Namen eingeben: ");
String name1 = sc.nextLine();
System.out.print("Bitte 2. Namen eingeben: ");
String name2 = sc.nextLine();
if (name1 == name2) {
  System.out.println("Die Namen sind identisch!");
} else {
  System.out.println("Die Namen sind unterschiedlich!");
}
sc.close();
```

Strings sind Objekte und sollten nicht mit den normalen Vergleichsoperatoren verglichen werden

Vergleich von Strings

Richtig

```
Scanner sc = new Scanner(System.in);
System.out.print("Bitte 1. Namen eingeben: ");
String name1 = sc.nextLine();
System.out.print("Bitte 2. Namen eingeben: ");
String name2 = sc.nextLine();
if (name1.equals(name2)) {
  System.out.println("Die Namen sind identisch!");
} else {
  System.out.println("Die Namen sind unterschiedlich!");
}
sc.close();
```

Aufgabe: Einfacher Rechner

GRUNDLAGE (PABS)

Es soll ein simpler Rechner programmiert werden, der zwei ganze Zahlen vom Benutzer einliest und danach folgende Ausgabe erzeugt:

```
Erste Zahl: 10
Zweite Zahl: 5

Summe: 15
Differenz: 5
Produkt: 50
Quotient: 2
Rest: 0
```

Falls die zweite Zahl 0 ist, können Quotient und Rest nicht gebildet werden. In diesem Fall sollen die entsprechenden Ausgaben durch „Kann nicht durch 0 teilen!“ bzw. „Kann keinen Rest gegen 0 bilden!“ ersetzt werden!

Logische Operatoren

Operator	Beispiel	Funktion	Trifft zu wenn ...
`&&`	`a && b`	logisches Und (Konjunktion)	... sowohl `a` als auch `b` wahr sind
`||`	`a || b`	logisches Oder (Disjunktion)	... entweder `a` oder `b` wahr sind, **oder beide**
`!`	`!a`	Nicht (Negation)	... `a` nicht wahr ist

Logisches Und

Beispiel aus der Praxis:

Wenn es draußen heiß ist
und die Sonne scheint,
dann gehe ich ins Freibad.

a	b	a && b
true	true	true
true	false	false
false	true	false
false	false	false

Beispiel als Java-Programm:

```
int temperature = 28;
boolean sonne = true;
if (temperature >= 20 && sonne == true) {
  System.out.println("Ich gehe ins Freibad!");
}
```

Logisches Oder

Beispiel aus der Praxis:

Wenn es nach 22 Uhr ist
oder ich müde bin,
dann gehe ich ins Bett.

a	b	a || b
true	true	true
true	false	true
false	true	true
false	false	false

Beispiel als Java-Programm:

```
int uhrzeit = 23;
boolean muede = true;
if (uhrzeit > 22 || muede) {
  System.out.println("Ich gehe ins Bett!");
}
```

Achtung:
Kein Exklusiv-Oder.
Wenn es nach 22 Uhr ist **und** ich gleichzeitig müde bin, trifft die Bedingung auch zu!

Negation

Beispiel aus der Praxis:

Wenn es nicht regnet,
dann kann ich den Regenschirm zuhause lassen.

a	!a
true	false
false	true

Beispiel als Java-Programm:

```
boolean regen = true;
if (!regen) {
  System.out.println("Ich kann den Regenschirm zuhause lassen!");
}
```

Aufgabe: Menü

GRUNDLAGE

Ziel ist es, ein Menü zu programmieren, das drei verschiedene Funktionen bietet. Die Funktionen seht ihr unten in der Beispielausgabe. Eine Menüoption soll durch Eingabe einer Kennzahl ausgewählt werden können.

```
1. Hallo ausgeben
2. Zahl quadrieren
3. Zahl größer 0 testen
4. "Programmende" ausgeben

Programmfunktion? 3
Welche Zahl soll geprüft werden? 43
Zahl ist größer 0!
```

ZUSATZ

- Schreibt ein kleines Programm, dass zwei Kommazahlen i und j und einen boolean b einließt und anschließend folgende Bedingungen prüft:
  - Wenn i > 5 **und** j < 5 ist, soll **"1"** ausgegeben werden.
  - Wenn **mindestens** eine Zahl negativ ist, soll **"2"** ausgegeben werden.
  - Wenn **entweder** i+j < 10 ist **oder** i*j nicht 0 ist, soll **"3"** ausgegeben werden.
  - Wenn keine der obigen Bedingungen erfüllt ist, soll **"4"** ausgegeben werden.
- Wenn b true ist, sollen alle Zahlen ausgegeben werden, deren Bedingung stimmt. Wenn b false ist, dann nur die erste, die zutrifft.

Fallunterscheidung

Fallunterscheidung mit “If/Else“ möglich aber umständlich:

```
short weekday = 1;
if (weekday == 1)
  System.out.println("Es ist Montag");
else if (weekday == 2)
  System.out.println("Es ist Dienstag");
else if (weekday == 3)
  System.out.println("Es ist Mittwoch");
else if (weekday == 4)
  System.out.println("Es ist Donnerstag");
else if (weekday == 5)
  System.out.println("Es ist Freitag");
else
  System.out.println("Es ist Wochenende");
```

Fallunterscheidung

Elegantere Lösung: Fallunterscheidung mit Switch-Case:

```
short weekday = 1;
switch (weekday) {
case 1:
  System.out.println("Es ist Montag");
  break;
case 2:
  System.out.println("Es ist Dienstag");
  break;
case 3:
  System.out.println("Es ist Mittwoch");
  break;
case 4:
  System.out.println("Es ist Donnerstag");
  break;
case 5:
  System.out.println("Es ist Freitag");
  break;
default:
  System.out.println("Es ist Wochenende");
  break;
}
```

Fallunterscheidung mit "Switch" um eine Variable gegenüber mehrerer Werte zu prüfen

"Case" um einen Wert zu prüfen und einen Fall zu beginnen

"Break" beendet einen Fall. Ohne "Break" wird der nächste Fall abgearbeitet, unabhängig vom Wert der Variable

"Default" wird verwendet, wenn kein Fall zutrifft. Der "Default"-Zweig ist optional.

Aufgabe: Menü 2.0

GRUNDLAGE

Schreibt die Abfrage nach der gewünschten Menüoption nun in eine Switch/Case-Fallunterscheidung um.

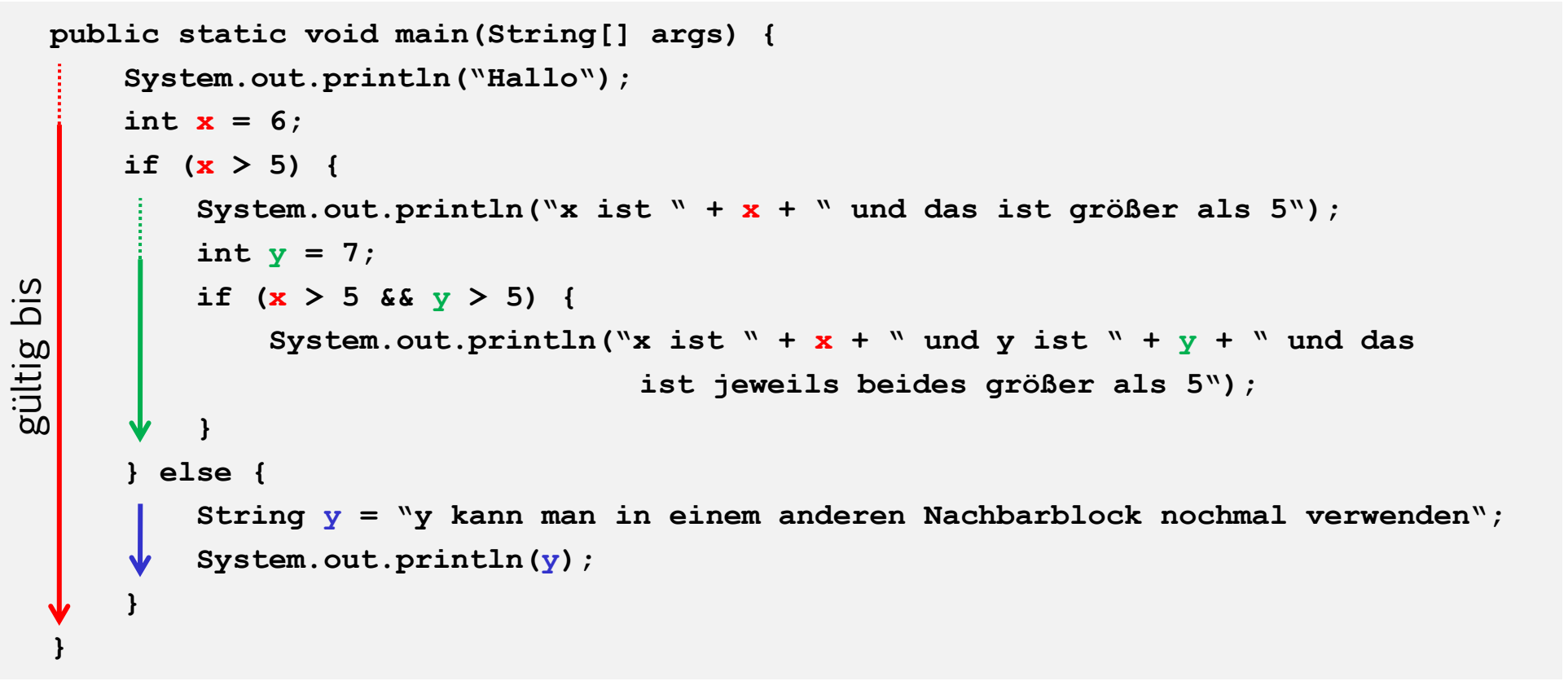
```
switch (eingabe) {
  case 1:
    // "Hallo" implementieren
    break;
  case 2:
    // "Quadrieren" implementieren
    break;
  case 3:
    // "Groesser als" implementieren
    break;
  default:
    // Ende
 }
```

ZUSATZ

- Schreibt ein Programm, dass eine positive Ganzzahl einließt und mittels switch case folgendes ausgibt:
  - "A" wenn Zahl < 2
  - "B" wenn Zahl < 4
  - "C" wenn Zahl < 6
  - "D" immer
- Wenn mehrere Bedingungen zutreffen, sollen alle Buchstaben ausgeben werden (Zahl 1 ergibt z.B. ABCD)

Gültigkeit von Variablen (1/2)

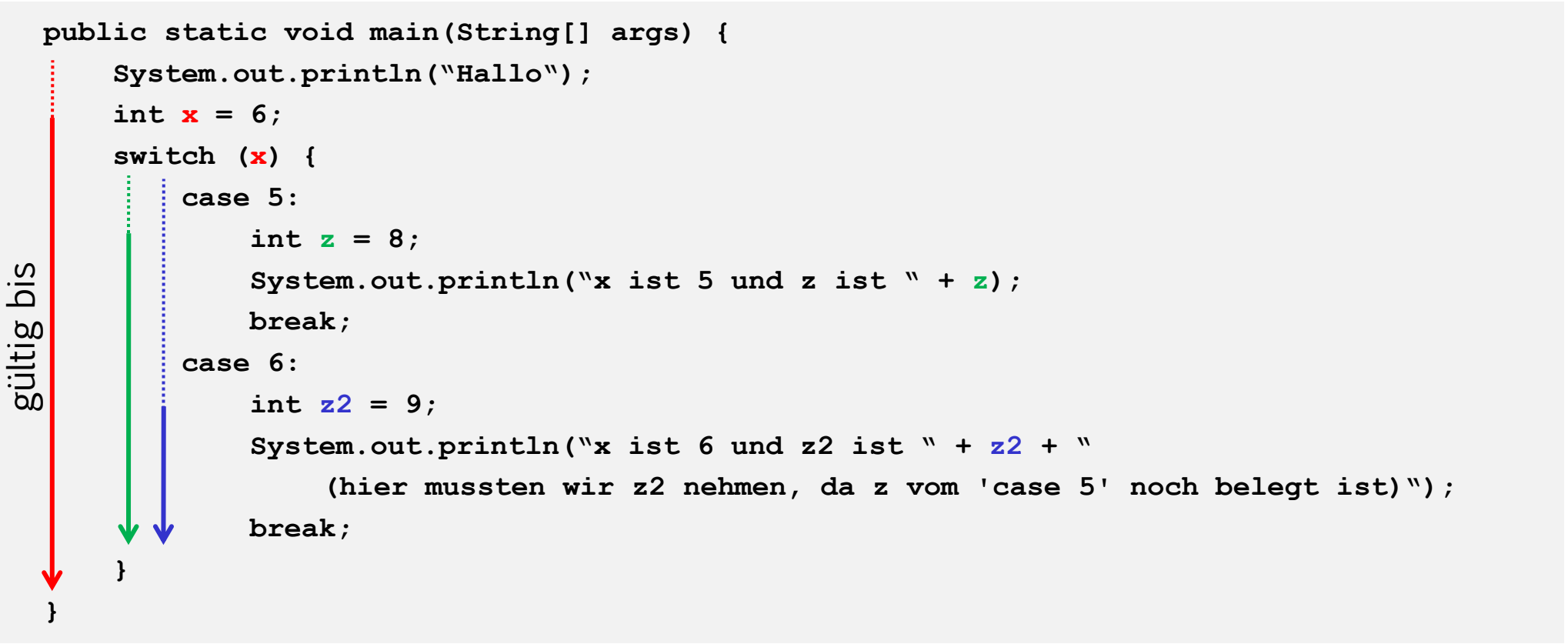
Variablen haben einen Gültigkeitsbereich, der sich auf den Bereich innerhalb der geschweiften Klammern erstreckt, in dem die Variable deklariert wurde.




```
public static void main(String[] args) {
    System.out.println("Hallo");
    int x = 6;
    if (x > 5) {
        System.out.println("x ist " + x + " und das ist größer als 5");
        int y = 7;
        if (x > 5 && y > 5) {
            System.out.println("x ist " + x + " und y ist " + y + " und das
                                ist jeweils beides größer als 5");
        }
    } else {
        String y = "y kann man in einem anderen Nachbarblock nochmal verwenden";
        System.out.println(y);
    }
}
```

Gültigkeit von Variablen (2/2)

Variablen haben einen Gültigkeitsbereich, der sich auf den Bereich innerhalb der geschweiften Klammern erstreckt, in dem die Variable deklariert wurde.




```
public static void main(String[] args) {
    System.out.println("Hallo");
    int x = 6;
    switch (x) {
        case 5:
            int z = 8;
            System.out.println("x ist 5 und z ist " + z);
            break;
        case 6:
            int z2 = 9;
            System.out.println("x ist 6 und z2 ist " + z2 + "
                (hier mussten wir z2 nehmen, da z vom 'case 5' noch belegt ist)");
            break;
    }
}
```

Warum Methoden

Befehle müssen immer wieder neu geschrieben werden.

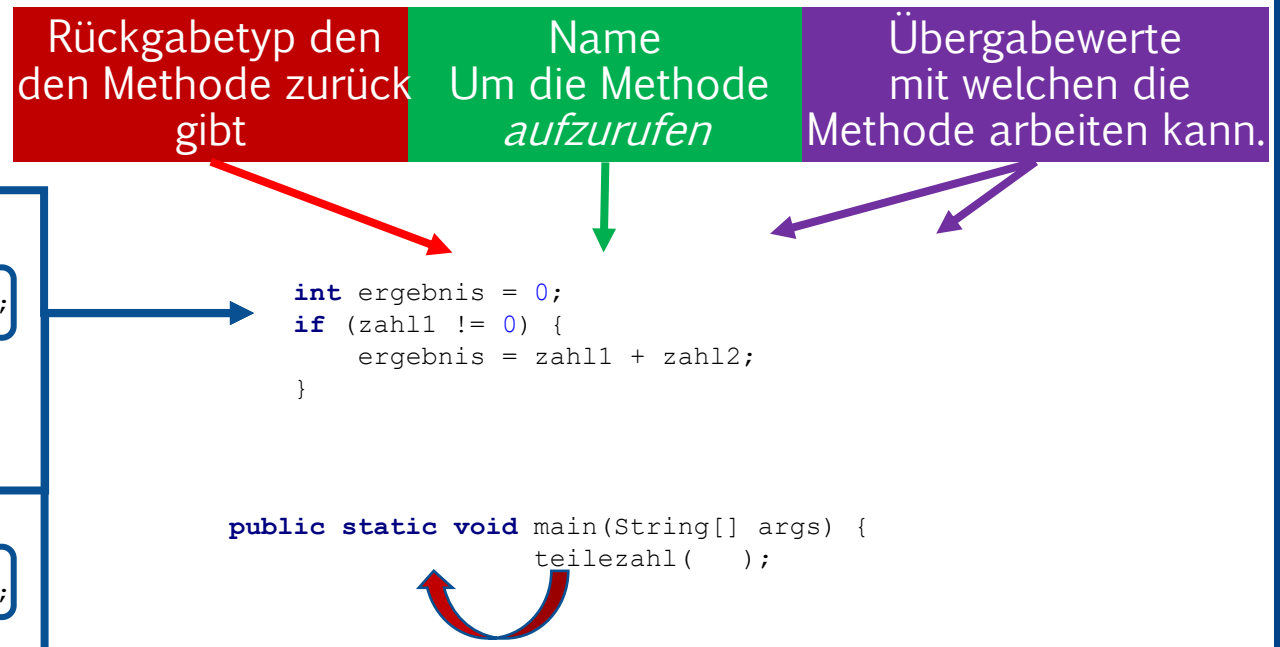
Code Blöcke mit gleicher Funktion müssen kopiert werden.

Dafür gibt es ***Methoden.***

```
public static void main(String[] args) {
    int zahl1 = 2;
    int zahl2 = 2;
    int ergebnis = 0;
    if (zahl1 != 0) {
        ergebnis = zahl1 + zahl2;
    }
    System.out.println("Das Ergebnis ist: " + ergebnis);

    zahl1 = 1;
    zahl2 = 4;
    if (zahl1 != 0) {
        ergebnis = zahl1 + zahl2;
    }
    System.out.println("Das Ergebnis ist: " + ergebnis);

    zahl1 = 1;
    zahl2 = 4;
    if (zahl1 != 0) {
        ergebnis = zahl1 + zahl2;
    }
    System.out.println("Das Ergebnis ist: " + ergebnis);
}
```

Methoden

```
public static void main(String[] args) {
    int zahl1 = 2;
    int zahl2 = 2;

    int ergebnis = 0;
    if (zahl1 != 0) {
        ergebnis = zahl1 / zahl2;
    }

    System.out.println("Das Ergebnis ist: " + ergebnis);

    zahl1 = 1;
    zahl2 = 4;

    if (zahl1 != 0) {
        ergebnis = zahl1 / zahl2;
    }

    System.out.println("Das Ergebnis ist: " + ergebnis);

    zahl1 = 0;
    zahl2 = 10;

    if (zahl1 != 0) {
        ergebnis = zahl1 / zahl2;
    }

    System.out.println("Das Ergebnis ist: " + ergebnis);
}
```



Aufgabe: Menü 2.0 mehrmals

GRUNDLAGE

Schreibt das Menü 2.0 so um, dass die Fallunterscheidung in einer Methode erfolgt.
Ruft diese fünf mal von *main* aus auf. Lest dabei jedes mal eine neue Eingabe ein.

```
public static void main(String[] args) {
  //...
  String eingabe = //Eingabe einlesen
  System.out.println(euereMethode(eingabe))
  //wiederholt es 5 mal
}
```

ZUSATZ

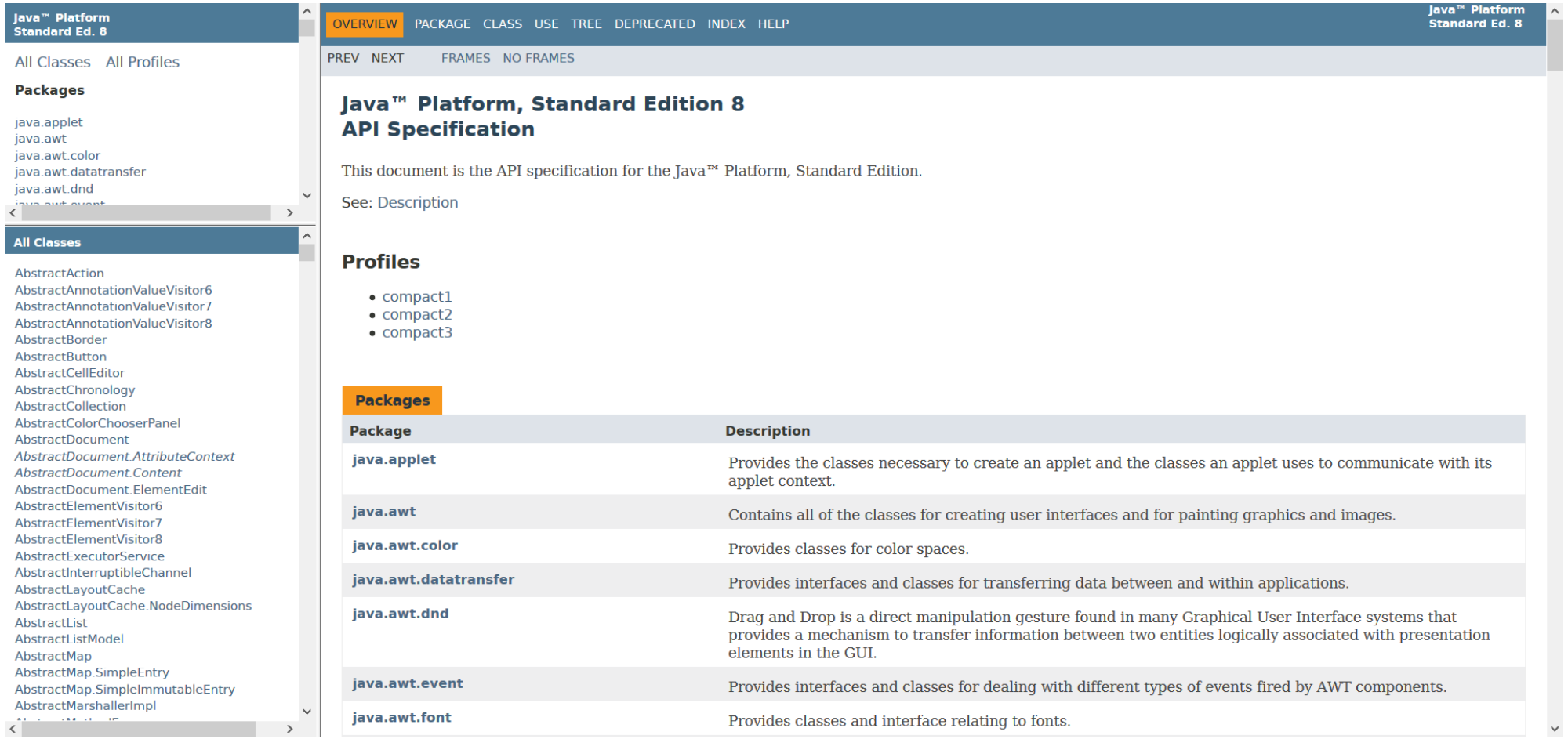
› Lasst auch die Eingabe über eine Methode erfolgen

Recherche – Java ist auch eine Insel

- http://openbook.rheinwerk-verlag.de/javainsel/

Recherche – Java API

- https://docs.oracle.com/javase/8/docs/api/



Übungen aus der Aufgabensammlung

Name	Seite	Schwierigkeit (1-10)	Freiheit (0-3)
Volumen/Oberfläche	9	2	1
Einlesen/Pi raten	10	2	1
Ascii/Strings	11	2	1
Reaktionsfähigkeit	30	8 (viel Recherche)	3

Abschlussaufgabe: Würfeln

Implementiert ein Würfelspiel, bei dem mit zwei Würfeln (1-6) gewürfelt wird. Zu Beginn des Spieles legt der Spieler einen Einsatz (Geldbetrag) fest. Je nach Augensumme der Würfel bekommt der Spieler eine Auszahlung, die der folgende Tabelle entnommen werden kann.

Augensumme	Auszahlung
12	4facher Einsatz
11	3facher Einsatz
10	2facher Einsatz
7, 8, 9	1facher Einsatz
2, 3, 4, 5, 6	keine

Lösungshinweis (Zufallszahlen generieren):

```
Random rnd = new Random();
int wuerfel1 = rnd.nextInt(6);
int wuerfel2 = rnd.nextInt(6);
```

Was muss geändert werden?

Gebt dabei sowohl Augensumme, Auszahlung und Gewinn an.