

# Brückenkurs Programmieren

## Tag 1: Variablen, Operatoren und Verzweigungen

Jakob Czekansky, M.Sc.

Technische Hochschule Mittelhessen  
25. März 2024



## Organisatorisches

Einleitung: Was ist Programmieren?

Hello World

## Variablen und Datentypen

- Variablen

- Operatoren

## Verzweigungen

- If-Abfrage

- Logische Ausdrücke

- Logische Operatoren

## Organisatorisches

Einleitung: Was ist Programmieren?

Hello World

Variablen und Datentypen

Variablen

Operatoren

Verzweigungen

If-Abfrage

Logische Ausdrücke

Logische Operatoren

## Ziele des Kurses:

- ▶ Grundzüge des Programmierens lernen (unabhängig von der Sprache)
- ▶ Spaß haben

## Themen:

- ▶ Heute: Variablen, Operatoren, Verzweigungen
- ▶ Dienstag: Animationen, Schleifen, Arrays
- ▶ Mittwoch: Funktionen, Events
- ▶ Donnerstag: Wert- vs. Referenzsemantik, Objektorientierung
- ▶ Freitag: Rückblick, Ausblick, Programmierumgebungen

## Arbeitsablauf:

- ▶ Beginn pünktlich um 9:00 Uhr im großen Hörsaal A20.1.36
- ▶ Kurzer Vorlesungsteil
  - ▶ Theorie mit gemeinsamen, kleinen Übungen
- ▶ Rest der Zeit:
  - ▶ Bearbeitung der Übungsblätter in zwei Übungsräumen

## Tagesablauf:

- ▶ Erster Block: 9:00 - 12:00 Uhr
- ▶ Mittagspause: 12:00 - 13:00 Uhr
- ▶ Zweiter Block: 13:00 - 16:00 Uhr

## Unterlagen:

- ▶ <https://brueckenkurs-programmieren.thm.de>

Organisatorisches

**Einleitung: Was ist Programmieren?**

Hello World

Variablen und Datentypen

Variablen

Operatoren

Verzweigungen

If-Abfrage

Logische Ausdrücke

Logische Operatoren

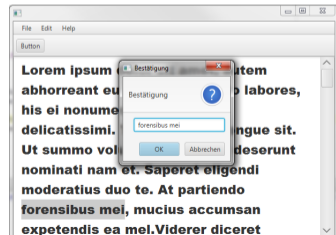
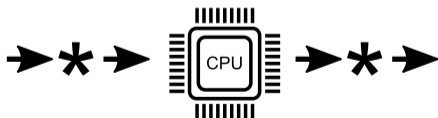
# Was ist Programmieren?

## Definition

Programmieren ist das Schreiben von Computerprogrammen.

Ein Computerprogramm ist eine in einer Programmiersprache verfasste Anweisungsabfolge, die ein bestimmtes Problem löst.

```
query = getUserInput();
text = textArea.getText();
if(text.contains(query)){
    textArea.highlight(query);
} else {
    showMessage("Not found.");
}
```



# Programmiersprachen

## Maschinensprache

```
0100 1101 0101 1010 1001 0000
0000 0000 0000 0011 0000 0000
0000 0000 0000 0000 0000 0100
```

## Assembler

```
pushc 3
pushc 4
add
```

## C

```
int main(int argc, char *argv[]) {
    char *str = (char*) malloc(10);
    str[9] = 0;
    printf("%s", str);
}
```

## Java

```
public class Thing {
    public static void main(String[] args){
        System.out.println(args.length);
    }
}
```

## Processing

```
int[] test = new int[10];
println(test.length);
```

## Ruby

```
data = FileIO.read("test.csv")
data = data.split("\n")
data = data.map {|x| x.split(";")}
```



# Klassifizierung von Programmiersprachen

## Paradigmen

- ▶ funktional (Lisp, Haskell)
- ▶ objektorientiert (Java)
- ▶ prozedural (C, Fortran)
- ▶ multi-paradigm (Ruby)

## Andere Unterklassen

- ▶ scripting language (Ruby, Python, JavaScript)
- ▶ domain-specific (VHDL, SPSS, Matlab)

## Eigenschaften

- ▶ memory safety
- ▶ strong/weak typing
- ▶ dynamic/static typing
- ▶ plattformunabhängig
- ▶ metaprogramming
- ▶ pattern matching

# Wie bringt man einem Computer eine Sprache bei?

---

Frage: Wie werden Programme in höheren Programmiersprachen ausführbar gemacht?

# Wie bringt man einem Computer eine Sprache bei?

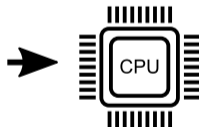
Frage: Wie werden Programme in höheren Programmiersprachen ausführbar gemacht?

Antwort: Mit einem Programm.

```
query = getUserInput();  
text = textArea.getText();  
if(text.contains(query)){  
    textArea.highlight(query);  
} else {  
    showMessage("Not found.");  
}
```

Interpreter / Compiler

```
0010 1101 1001 1010  
1001 1001 1100 1101  
1010 1010 1010 1111  
1000 0000 1010 1000  
1000 0111 0001 0010  
1010 1101 1101 0001  
0000 0000 1111 0111  
0101 1010 0101 1011  
1101 1101 0001 1111
```



Organisatorisches

Einleitung: Was ist Programmieren?

**Hello World**

Variablen und Datentypen

Variablen

Operatoren

Verzweigungen

If-Abfrage

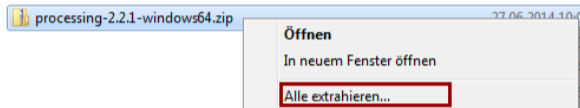
Logische Ausdrücke

Logische Operatoren

# Hello World in Processing

- ▶ Processing herunterladen: <https://processing.org/download>

- ▶ Zip-Archiv extrahieren



- ▶ in sinnvollen Ordner (z.B. C:\Programme\Processing)
- ▶ processing.exe starten
- ▶ Folgenden Code eingeben und Play-Button drücken:

```
ellipse(50,50,25,25);
```

# Erste Herausforderung: Ein Smiley

Aufgabe: Zeichne ein Smiley-Gesicht mit Processing.



## Hilfreiche Processing-Funktionen

```
size(w,h);  
ellipse(x,y,w,h);  
rect(x,y,w,h);  
background(c);  
line(x1,y1,x2,y2);  
fill(r,g,b);  
stroke(r,g,b);  
strokeWeight(w);
```

Mehr unter [processing.org/reference](https://processing.org/reference)

Organisatorisches

Einleitung: Was ist Programmieren?

Hello World

**Variablen und Datentypen**

Variablen

Operatoren

Verzweigungen

If-Abfrage

Logische Ausdrücke

Logische Operatoren

# Variablen: Wozu?

---

Was müsstest du tun, um das Gesicht nach rechts oder links zu verschieben?



# Variablen: Wozu?

---

Was müsstest du tun, um das Gesicht nach rechts oder links zu verschieben?

- ▶ Idee 1: Alle Werte von Hand anpassen
  - ▶ ca. 7 Zahlen

# Variablen: Wozu?

---

Was müsstest du tun, um das Gesicht nach rechts oder links zu verschieben?

- ▶ Idee 1: Alle Werte von Hand anpassen
  - ▶ ca. 7 Zahlen
- ▶ Idee 2: Gesichtsmittelpunkt  $(x,y)$  in **Variablen** speichern
  - ▶ linkes Auge befindet sich jetzt z.B. immer an Position  $(x-40,y-20)$
  - ⇒ nur noch  $x$  und  $y$  anpassen, statt allen Zahlen

# Variablen: Wozu?

Was müsstest du tun, um das Gesicht nach rechts oder links zu verschieben?

- ▶ Idee 1: Alle Werte von Hand anpassen
  - ▶ ca. 7 Zahlen
- ▶ Idee 2: Gesichtsmittelpunkt (x,y) in **Variablen** speichern
  - ▶ linkes Auge befindet sich jetzt z.B. immer an Position (x-40,y-20)
  - ⇒ nur noch x und y anpassen, statt allen Zahlen

## Processing

```
float x = 50;  
float y = 50;
```

# Aufgabe: Wandersmiley

## Vorgabe

```
float x = 50; //Werte dürfen verändert werden  
float y = 50;
```

Schreibe deinen Smiley-Code so um, dass du das Gesicht durch Verändern der Variablen x und y verschieben kannst.



# Variablen: Variablen als Behälter



Behälter, Typ: Flüssigkeit, Wert: 0.0 ml



Behälter, Typ: Stifte, Wert: 11

# Variablen: Definition

## Definition

Eine Variable gibt einer **Speicherstelle** einen **Namen**. Sie kann nur Werte entsprechend ihres **Typs** enthalten.

## Syntax: Variablendeklaration

```
<Typ> <Name> = <Wert>;
```

## Beispiel

```
float x = 3.5;
```

# float

- ▶ Fließkommazahlen (engl.: *floating point number*)
- ▶ Punkt statt Komma
- ▶ 4 Byte in IEEE754-Codierung  $\Rightarrow$  ca. 7 geltende Stellen
  - ▶ 0.0001234567
  - ▶ 1234567.0
  - ▶ 1234567000.0
  - ▶ ~~123456789.012~~ (wird gerundet)
  - ▶ 1e127 ( $1 \cdot 10^{127}$ )
  - ▶ 2.3e-128 ( $2.3 \cdot 10^{-128}$ )

## int

- ▶ Ganzzahlen (engl.: *integer*)
- ▶ 4 Byte in Zweierkomplementdarstellung
  - ▶ 4
  - ▶ -7
  - ▶ 0
  - ▶ 2147483647 ( $2^{31} - 1$ )
  - ▶ 3147483647
  - ▶ -2147483648 ( $-2^{31}$ )



## boolean

- ▶ Wahrheitswerte (nach George Bool)
- ▶ exakt zwei mögliche Werte
  - ▶ `true`
  - ▶ `false`

# String

- ▶ Zeichenketten (engl.: *string*)
- ▶ Zusammengesetzter Typ aus mehreren Zeichen (`char`)
  - ▶ "Blabla"
  - ▶ "123"
- ▶ Länge beliebig

Wie erhöhe ich den Wert einer Zahlvariable x um 2?

```
x = x + 2;
```

## Definition

Die Rechenzeichen +, -, \* und / nennt man **arithmetischen Operatoren**. Sie führen mathematische Rechenoperationen aus.

## Definition

Das einfache Gleichzeichen = ist der **Zuweisungsoperator**. Er weist der Variable auf der linken Seite den Wert des Ausdrucks auf der rechten Seite zu.

## Organisatorisches

Einleitung: Was ist Programmieren?

Hello World

## Variablen und Datentypen

Variablen

Operatoren

## Verzweigungen

If-Abfrage

Logische Ausdrücke

Logische Operatoren

# Verzweigungen

Immer nur das Gleiche tun ist langweilig.

- ▶ Bisher: Aneinanderreihung von einzelnen Befehlen
- ▶ Wie im Computerspiel: Unterschiedliche Pfade machen Code interessant

## Idee

Je nach Wert einer `boolean`-Variablen zwinkert das Smiley oder nicht.



# Verzweigungen: If-Abfrage

## Syntax

```
if (<Bedingung>) {  
    <Anweisungsblock1>  
} else {  
    <Anweisungsblock2>  
}
```



## Definition

Eine **If-Abfrage** entspricht einer Weiche.

- ▶ Bedingung ist wahr (`true`): Führe **If-Teil** (Block 1) aus.
- ▶ Bedingung ist falsch (`false`): Führe **Else-Teil** (Block 2) aus.

# Verzweigungen: Beispiele für If-Abfragen

## Beispiel 1

```
if (x == 0) {  
    println("Du erzeugst ein schwarzes Loch!");  
} else {  
    println("4 geteilt durch " + x + " ist " + (4/x));  
}
```

## Beispiel 2

```
if (x < 0) {  
    x = -x;  
}  
  
//Else-Teil darf auch wegfallen
```

# Aufgabe: Zwinkernder Smiley

Baue eine `boolean`-Variable in deinen Smiley-Code ein, die bestimmt, ob der Smiley zwinkert oder nicht.



## Erinnerung: If-Syntax

```
if (<Bedingung>) {  
    <Anweisungsblock1>  
} else {  
    <Anweisungsblock2>  
}
```



# Verzweigungen: Bedingungen / Logische Ausdrücke

Was kann man eigentlich alles als Bedingung in einem If verwenden?

## Definition

Bedingungen sind **logische Ausdrücke**, das heißt Ausdrücke, die einen Wahrheitswert ergeben (`true` oder `false`).

## Beispiele: true

```
3 < 4
4 <= 4
1 == 1
2+1 != 4
true
```

## Beispiele: false

```
3 > 4
5 <= 4
1 == 2
2+1 != 4-1
false
```

## Definition

**Logische Operatoren** errechnen einen Wahrheitswert aus anderen Werten (und bilden damit logische Ausdrücke).

`==`

`<`

`<=`

`!=`

`>`

`>=`

# Verzweigungen: Verknüpfungen von logischen Operatoren

## Logischer Ausdruck

Der Wert von  $x$  liegt zwischen 0 und 10.

# Verzweigungen: Verknüpfungen von logischen Operatoren

## Logischer Ausdruck

Der Wert von  $x$  ist größer als 0 und kleiner als 10.

# Verzweigungen: Verknüpfungen von logischen Operatoren

## Logischer Ausdruck

Der Wert von  $x$  ist größer als 0 und kleiner als 10.

Um kompliziertere Bedingungen auszudrücken gibt es auch Operatoren, die logische Ausdrücke **negieren** oder miteinander **verknüpfen** können:

# Verzweigungen: Verknüpfungen von logischen Operatoren

## Logischer Ausdruck

Der Wert von x ist größer als 0 und kleiner als 10.

Um kompliziertere Bedingungen auszudrücken gibt es auch Operatoren, die logische Ausdrücke **negieren** oder miteinander **verknüpfen** können:

a && b

a und b

a || b

a oder b

!a

nicht a

# Zusammenfassung

## Variablen

```
int i = 10;  
float f = 3.5;  
boolean b = true;  
char c = 'x';  
String blabla = "Blabla";
```

## Logische Operatoren

>   >=   ==   !=  
&&   ||   !

## If-Abfrage

```
if (<Bedingung>) {  
    <Anweisungsblock1>  
} else {  
    <Anweisungsblock2>  
}
```

## Arithmetische Operatoren

+   -   \*   /   \%