

# Listen

Foliensatz 2

Jasmin Noll  
wi20086@lehre.dhbw-stuttgart.de

21.04.2022

# Was euch heute erwartet

- Listen
  - Konventionen
  - Auf Elemente zugreifen und ändern
  - Elemente anhängen und einfügen
  - Elemente löschen
  - Sortieren
  - Slicing

# Listen

# Konventionen

- Eine Liste kann Elemente von jedem Datentyp enthalten
- Ein Element kann mehrfach vorkommen
- Variablennamen im Plural
  - Beispiel: `name` < `names`
- Listen werden mit `[ ]` erstellt und ausgegeben
  - Beispiel: `["Element 0", "Element 1", "Element 2"]`

# Auf Elemente zugreifen und ändern

- Zugriff auf ein Element mit `[ ]` und den Index des Elements
  - Beispiel: `names[0]`
- Um ein Element zu ändern wird es (wie bei einem Zugriff) ausgewählt und der neue Wert wird übergeben
  - Beispiel: `names[0] = "neuer Wert"`

<b>H</b>	<b>A</b>	<b>L</b>	<b>L</b>	<b>O</b>
0	1	2	3	4
-5	-4	-3	-2	-1

# Elemente anhängen und einfügen

- Mit `.append()` kann man ein Element ans Ende einer Liste hängen
  - Beispiel: `names.append("Max Mustermann")`
- Mit `.extend()` kann man mehrere Elemente einzeln ans Ende einer Liste hängen
  - Beispiel: `names.extend(list_of_names)`
- Mit `.insert()` kann man ein Element an einer bestimmten Stelle einfügen
  - Dafür gibt man erst den Index und dann das Element der Funktion mit
  - Beispiel: `names.insert(2, "Max Mustermann")`

# Elemente löschen

- Mit `.remove()` kann man ein bestimmtes Element entfernen
  - Dafür wird das Element benötigt
  - Beispiel: `names.remove("Max Mustermann")`
- Mit `.pop()` kann man ein Element nach einem bestimmten Index entfernen
  - Wird kein Index angegeben, so wird das letzte Element der Liste entfernt
  - Das Element wird parallel zum entfernen dieses nochmal aus
  - Beispiel: `names.pop(3)`
- Mit `del` kann ebenfalls ein bestimmtes Element mittleres Index oder die gesamte Liste an sich löschen
  - Beispiele:
    - `del names`
    - `del names[0]`

# Sortieren

- Mit `.sort()` können Listen alphabetisch oder numerisch sortiert werden
  - Der Standard ist hier die aufsteigende Sortierung. Mit dem Parameter `reverse = True` kann man absteigend sortieren
  - Beispiele:
    - `names.sort()`
    - `numbers.sort(reverse = True)`
- Mit `.reverse()` kann man die Liste einmal umsortieren, also die Reihenfolge der Elemente werden vertauscht
  - Beispiel: `names.reverse()`



# Slicing

- `[start index:end index]`
- Die Ergebnis endet ein Element vor *end index* und ist damit nicht mehr im Ergebnis enthalten
- Beispiele:
  - `variable[0:3]` → Ergebnis: Elemente 0, 1 und 2
  - `names[:4]` → Ergebnis: Elemente von start bis inkl. Element 3
  - `names[4:]` → Ergebnis: Elemente von Element 4 bis zum Ende
  - `names[:]` → Ergebnis: Alle Elemente

<b>H</b>	<b>A</b>	<b>L</b>	<b>L</b>	<b>O</b>
0	1	2	3	4
-5	-4	-3	-2	-1

# Übungen

