Grundlagen der Programmierung für Nicht-Informatiker

Aufgabensammlung, Tag 1

Giuseppe Accaputo

g@accaputo.ch

Aufbaben zu:
Variablen, Anweisungen, Ausdrücke, und alles dazwischen

Aufgabe 1.1: Ein erstes Programm

- Schreibe ein Python Programm, dass drei Variablen verwendet: `stadt, land, fluss`
- Weise allen drei Variablen einen passenden String zu
- Gib zuerst `Im folgenden steht eine Stadt, ein Land, und ein Fluss` auf einer Zeile aus
- Auf den nächsten Zeilen gibst Du die drei Variablen aus (pro Variable eine Zeile)

- Mögliche Ausgabe auf dem Bildschirm:

BILDSCHIRM AUSGABE

```
Im folgenden steht eine Stadt, ein Land, und ein Fluss
Zürich
Schweiz
Limmat
```

Aufgabe 1.2: Durchschnitt berechnen

- Schreibe ein Python Programm, dass den Durchschnitt aus den vier Werten 4, 5, 6, und 9 berechnet
- Verwende für jeden Wert eine eigene Variable
- Speichere das Resultat in eine eigene Variable
- Gib das Resultat danach aus

Aufgabe 1.3: Benutzereingaben ausgeben

- Schreibe ein Python Programm, dass nach den Namen des Benutzers fragt und diesen wieder ausgibt
- *Tipp*: `eingabe = input()` fragt den Benutzer nach einer Eingabe und speichert diese direkt in die Variable `eingabe` ab

- Mögliche Ausgabe auf den Bildschirm

BILDSCHIRM AUSGABE

```
Bitte gib Deinen Namen ein: Giuseppe
Hallo, Giuseppe!
```

Aufgabe 1.4: Fehlersuche

– Versuche alle Fehler im folgenden Code zu finden (ohne PyCharm wenn möglich):

CODE

```
auto_marke = 'Audi'
modell_name = 'A'
1modell_nr = 1
print(auto_marke + ' ' + Modell_name + ' ' + modell_nr)
```

Aufgabe 1.5: Weitere Lohnwerte berechnen

- Wir möchten ein Python-Programm schreiben, dass uns einige Lohnwerte berechnet
- Unser aktueller Stundenansatz beträgt aktuell 31.5 CHF
- Pro Tag arbeiten wir 8.4 Stunden
- Das Programm soll nun folgende Werte berechnen und ausgeben:
  1. Wieviel beträgt der Tageslohn?
  2. Wieviel beträgt der Monatslohn?
     - *Bemerkung*: Ein Monat besteht aus 20 Arbeitstagen

Universität Zürich[UZH]

Zentrale Informatik – IT Fort- und Weiterbildungen

Aufgaben zu:
Funktionen Teil 1 – Ein Einstieg

Aufgabe 2.1: Summe berechnen

- Definiere eine Funktion `summe`, welche die Summe bestehend aus drei Zahlen berechnet und ausgibt
- Rufe die Funktion anschliessend auf

- In der folgenden Tabelle findet ihr einige Funktionsaufrufe und die dazugehörigen Ausgaben um euer Programm auf dessen Korrektheit zu überprüfen:

Aufruf der Funktion im Programm	Mögliche Ausgabe auf dem Bildschirm
`summe(1,2,3)`	`6`
`summe(1,-1,0)`	`0`
`summe(-1,-2,-3)`	`-6`

- **Wichtig**: Die Funktion muss für beliebige Zahlen funktionieren

Aufgabe 2.2: Wurzel berechnen

- Definiere eine Funktion `wurzel`, welche die Wurzel aus dem Produkt bestehend aus zwei positiven oder zwei negativen Zahlen berechnet und ausgibt
- Rufe die Funktion anschliessend auf

- In der folgenden Tabelle findet ihr einige Funktionsaufrufe und die dazugehörigen Ausgaben um euer Programm auf dessen Korrektheit zu überprüfen:

Aufruf der Funktion im Programm	Mögliche Ausgabe auf dem Bildschirm
`wurzel(2,2)`	`2.0`
`wurzel(-3,-3)`	`3.0`
`wurzel(1,2)`	`1.4142135`

- **Wichtig**: Die Funktion muss für beliebige Zahlen funktionieren

Aufgaben zu:
Bedingte Anweisungen («Conditionals»)

Aufgabe 3.1: Zahlen vergleichen

– Schreibe eine Funktion `vergleich`, welche zwei Zahlen entgegennimmt und dabei ausgibt, ob die erste Zahl grösser, kleiner, oder gleich der zweiten Zahl ist

– In der folgenden Tabelle findet ihr einige Funktionsaufrufe und die dazugehörigen Ausgaben um euer Programm auf dessen Korrektheit zu überprüfen:

Aufruf der Funktion im Programm	Mögliche Ausgabe auf dem Bildschirm
`vergleich(1,2)`	`Erste Zahl ist kleiner als zweite Zahl`
`vergleich(100,2)`	`Erste Zahl ist grösser als zweite Zahl`
`vergleich(123,123)`	`Beide Zahlen sind gleich gross`

– **Wichtig**: Die Funktion muss für beliebige Zahlen funktionieren

Aufgabe 3.2: Wurzel berechnen – Teil 2

- Passe die Funktion `wurzel` aus Aufgabe 2.3 so an, dass nur ein Ergebnis ausgegeben wird, wenn die Multiplikation beider Zahlen ein positives Ergebnis ergibt
- Gib im Falle einer ungültigen Eingabe eine Fehlermeldung aus

- In der folgenden Tabelle findet ihr einige Funktionsaufrufe und die dazugehörigen Ausgaben um euer Programm auf dessen Korrektheit zu überprüfen:

Aufruf der Funktion im Programm	Mögliche Ausgabe auf dem Bildschirm
`wurzel(-4,4)`	`Fehler: Produkt ist negativ`
`wurzel(-3,-3)`	`3`
`wurzel(1,2)`	`1.4142135`

- **Wichtig**: Die Funktion muss für beliebige Zahlen wie erwartet funktionieren

Aufgabe 3.3: Sandwich-Funktion

– Schreibe eine Funktion `ist_sandwich`, die drei Zahlen `x,y,` und `z` entgegennimmt und `Ist ein Sandwich` ausgibt, wenn `x <= y <= z` erfüllt ist; ansonsten soll `Ist leider kein Sandwich` ausgegeben werden

– In der folgenden Tabelle findet ihr einige Funktionsaufrufe und die dazugehörigen Ausgaben um euer Programm auf dessen Korrektheit zu überprüfen:

Aufruf der Funktion im Programm	Mögliche Ausgabe auf dem Bildschirm
`ist_sandwich(3,1,2)`	`Ist kein Sandwich`
`ist_sandwich(1,3,2)`	`Ist kein Sandwich`
`ist_sandwich(1,2,3)`	`Ist ein Sandwich`

– **Wichtig**: Die Funktion muss für beliebige Zahlen funktionieren

Aufgabe 3.4: Laufen, Auto fahren, oder fliegen

- Schreibe ein Programm, dass den Benutzer fragt wie weit er reisen möchte (in Kilometer)
- Falls er weniger als 3.2 km reisen möchte, so soll das Programm ihm vorschlagen, den Weg zu laufen
- Falls er mehr als 3.2 km und weniger als 100 km reisen möchte, so soll das Programm ihm vorschlagen, bis ans Ziel mit dem Auto zu fahren
- Falls er mehr als 100 km reisen möchte, so soll das Programm vorschlagen die Strecke mit dem Flieger zu durchreisen

- *Tipp*: `eingabe = input()` speichert die Eingabe des Benutzers automatisch als String ab, d.h. wenn der Benutzer eine Zahl eingibt, so müssen wir eine Typumwandlung durchführen:

CODE

```
eingabe = input()
strecke_in_km = int(eingabe)
```