Hello World!

1) Download und Installation

Die Verwendung des Textes „Hello World!“ ist eine Tradition und geht auf Programming in C – A Tutorial zurück, ein internes Programmierhandbuch der Bell Laboratories über die Programmiersprache C, das Brian Kernighan 1974 verfasste.

Heute ist ein simples “Hello World” Programm die üblichste Methode um zu testen, ob die Programmierumgebung (Editor, Interpreter, Compiler, IDE,...) funktioniert.

Ein “Hello World” Programm in Python zu schreiben ist ganz einfach. Man braucht dazu bloß einen Python Interpreter und einen Text Editor (z.B. Atom oder Notepad++).

Es gibt Tools, die das Programmieren noch lustiger und einfacher machen: IDEs oder Integrated Development Environments (z.B. IDLE oder PyCharm

Eine Liste verschiedener Text Editors und IDEs findest du zum Beispiel hier.

```
1  print('Hello World!')
```

```
Hello World!
```

Versuche es selbst!

Hello World:

Erstelle eine neue Python-Datei. Benenne sie so: 1_HelloWorld_DeinVorname.py

Schreibe folgendes in die Datei:

print('Hello World')

Führe die Datei aus!

2) Text ausgeben mit print()

```
2  print('Eine kleine Eule')
   print('   (((')
   print('  (. .)')
   print(' (( v ))')
   print('---m-m---')
```

```
Eine kleine Eule
   (((
  (. .)
```

```
 (( v ))
---m-m---
```

3
```
print('So geht es einfacher:')
print('''
   (((
  (. .)
 (( v ))
---m-m---
''')
```

```
So geht es einfacher:

   (((
  (. .)
 (( v ))
---m-m---
```

4
```
print('''
Ein Hund:
0___/
 ||||
''')
```

```
Ein Hund:
0___/
 ||||
```

5
```
print('Eine Perlenkette:')
print('-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-')
```

```
Eine Perlenkette:
-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-
```

6
```
print('So geht es schneller:')
print('-o'*10+'-')
```

```
So geht es schneller:
-o-o-o-o-o-o-o-o-o-o-
```

7
```
print('Ein schönes Muster:')
print(' /\\ '*15)
print('/  \\'*15)
```

```
Ein schönes Muster:
 /\  /\  /\  /\  /\  /\  /\  /\  /\  /\  /\  /\  /\  /\  /\
/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \
```

Im Interaktiven Modus ist der print() Befehl nicht unbedingt notwendig:

8
```
'So geht es auch.'
```

8
```
'So geht es auch.'
```

9
```
So aber nicht.
```

```
  File "/var/folders/tt/z9q9vz1x0y907tzlmjs8tvrr0000gn/T/ipykernel_16918/2859723206.py", line
    So aber nicht.
```

```
      ^
SyntaxError: invalid syntax
```

Versuche es selbst!

Acii-Art:

Erstelle eine neue Python-Datei. Benenne sie so: 2_AsciiArt_DeinVorname.py

"Male" ein Tier mit dem print() Befehl.

Suche dir eine Vorlage (z.B. hier: https://www.asciiart.eu/animals) oder werde selbst kreativ.

```
10 print('''
                              _,add8ba,
                            ,d888888888b,
                           d8888888888888b                      _,ad8ba,_
                          d888888888888888)                   ,d888888888b,
                          I8888888888888888 _________        ,8888888888888b
                 __________`Y88888888888888P""""""""""baaa,__ ,888888888888888,
             ,adP"""""""""""9888888888P"""^               ^""Y8888888888888888I
          ,a8"^          ,d888P"888P^                        ^"Y8888888888P'
        ,a8^           ,d8888'                                 ^Y8888888P'
      a88'           ,d8888P'                                    I88P"^
    ,d88'          d88888P'                                      "b,
   ,d88'          d888888'                                        `b,
  ,d88'          d888888I                                         `b,
  d88I           ,8888888'              ___                        `b,
 ,888'           d8888888          ,d88888b,              ____       `b,
 d888           ,8888888I         d88888888b,           ,d8888b,      `b
,8888           I8888888I        d8888888888I          ,88888888b       8,
I8888           88888888b       d88888888888'          8888888888b      8I
d8886           888888888       Y888888888P'           Y8888888888,     ,8b
88888b          I88888888b      `Y8888888^             `Y888888888I     d88,
Y88888b         `888888888b,      `"""""^               `Y8888888P'    d888I
`888888b         88888888888b,                          `Y8888P^     d88888
 Y888888b       ,8888888888888ba,_          _______       `"""^     ,d888888
 I8888888b,    ,888888888888888888ba,_     d88888888b               ,ad8888888I
 `888888888b,  I88888888888888888888b,     ^"Y888P"^      ____.,ad88888888888I
  88888888888b,`888888888888888888888b,      ""      ad888888888888888888888'
  888888888888698888888888888888888888b_,ad88ba,_,d88888888888888888888888
  8888888888888888888888888888888888888b,`"""^ d888888888888888888888888I
  88888888888888888888888888888888888888baaad8888888888888888888888888888'
  Y888888888888888888888888888888888888888888888888888888888888888888888P
  I888888888888888888888888888888888888888P^  ^Y88888888888888888888'
  `Y88888888888888P8888888888888888888888'     ^888888888888888888I
   `Y8888888888888 `88888888888888888888       8888888888888888P'
    `Y888888888888  `8888888888888888888,     ,88888888888888P'
     `Y88888888888b  `88888888888888888888I     I88888888888888'
       "Y8888888888b  `88888888888888888888I     I8888888888888'
         "Y88888888P   `888888888888888888b     d888888888888'
            ^""""""^     `Y8888888888888888,    888888888888P'
                  Normand    "88888888888888888b,   Y888888888P^
                  Veilleux    `Y888888888888888b   `Y8888888P"^
                                "Y88888888888888P     `"""""^
                                  `"YY88888888888P'
                                       ^""""""""'
''')
```

```
                              _,add8ba,
                            ,d888888888b,
```

3) Kommentare

```
'''
Kommentare sind wichtig, damit du später noch weißt
was genau in welchem Abschnitt passiert.
Außerdem können so auch andere deinen Code besser verstehen.
Es gibt zwei verschiedene Arten von Kommentaren:
Lange Kommentare, die über mehrere Zeilen gehen
schreibst du am besten zwischen je drei Anführungszeichen.
'''

# Kurze Kommentare schreibst du nach einem # Symbol.
'''
Am Beginn eines neuen Scripts kann man einen Header einfügen.
Dieser Header beinhaltet meistens den Autor,
das Datum und eine Beschreibung des Inhalts.
'''

'''
author: Christine Ackerl
date: 29.03.2021

Einführung in Python:
```

```
print(), input(), f-string, +, -, *, /, **, %
Kommentare, Variablen, Datentypen
'''

# Programme lassen sich besser lesen, wenn Codezeilen kürzer sind.
# Du solltest also möglichst vermeiden, sehr lange Kommentare zu schreiben, die über die Fens
```

```
11 '\nauthor: Christine Ackerl\ndate: 29.03.2021\n\nEinführung in Python:\nprint(), input(), f-s
```

Versuche es selbst!

Kommentare:

Füge der Datei die du in Aufgabe 2 erstellt hast einen Header hinzu: Der Header sollte folgendes beinhalten: Author, Datum, Beschreibung und Link und Author der Vorlage (falls du eine Vorlage benutzt hast)

4) Variablen

```
12 a = 10
b = 2
c = a + b
print(c)
```

```
12
```

Variablen sollten einen aussagekräftigen Namen haben

```
13 einezahl = 35
nocheinezahl = 16
print(einezahl + nocheinezahl) # Befehle können kombiniert werden
```

```
51
```

Die Namen von Variablen können sogar Zahlen beinhalten:

```
14 Zahl1 = 1
print(Zahl1)
```

```
1
```

Eine Zahl alleine kann aber keine Variable sein:

```
15 1 = 2
print(1)
```

```
  File "/var/folders/tt/z9q9vz1x0y907tzlmjs8tvrr0000gn/T/ipykernel_16918/1794436354.py", line
    1 = 2
    ^
SyntaxError: cannot assign to literal
```

Warum funktioniert das hier nicht?

16
```
print(x)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)

/var/folders/tt/z9q9vz1x0y907tzlmjs8tvrr0000gn/T/ipykernel_16918/3546069579.py in <module>
----> 1 print(x)
      2

NameError: name 'x' is not defined
```

Typen von Variablen - Data types

17
```
a = 1
b = '1'
print (a,b)
```

```
1 1
```

auf den ersten Blick gibt es keinen Unterschied zwischen a und b. Versuchen wir folgendes:

18
```
print(a + b)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)

/var/folders/tt/z9q9vz1x0y907tzlmjs8tvrr0000gn/T/ipykernel_16918/2287542798.py in <module>
----> 1 print(a + b)
      2

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

warum funktioniert das nicht? Wir können den Typ der Variable abfragen:

19
```
print('Variable a ist ein')
print(type(a)) # Beim Kombinieren von Befehlen müssen wir auf die Klammern achten
print('Variable b ist ein')
print(type(b))
```

```
Variable a ist ein
<class 'int'>
Variable b ist ein
<class 'str'>
```

'str' steht für string = Kette/Schnur und wird für Zeichenabfolgen benutzt. Python kann strings speichen, ausgeben, und auch manipulieren, aber 'versteht' nicht was sie bedeuten. Python interpretiert Zeichen zwischen Anführungsstrichen als string.

'int' steht für integer = ganze Zahl. Mit integers kann man rechnen und vieles mehr. Python interpretiert 'einfache' Zahlen ohne Komma und ohne Anführungszeichen automatisch als integer.

Neben integers gibt es noch andere Typen von Zahlen:

```
20 x = 10.2
   print(type(x))
```

```
<class 'float'>
```

```
21 x = 5+3j
   print(type(x))
```

```
<class 'complex'>
```

Neben Zahlen und strings gibt es auch noch andere Arten von Daten:

```
22 z = True
   print(type(z))
```

```
<class 'bool'>
```

Später werden wir noch mehr Datentypen kennenlernen.

Data types umwandeln:

```
23 a = 10.5
   b = int(a)
   print(b)
```

```
10
```

```
24 a = '510.15'
   b = float(a)
   print(b)
```

```
510.15
```

5) Rechnenoperationen

```
25 print(10 + 35) # Addition
```

```
45
```

```
26 print(198 - 235) # Subtraktion
```

```
-37
```

```
27 print(17.5 * 2) # Multiplikation
```

```
35.0
```

```
28 print(25 / 5) # Division
```

```
5.0
```

```
29 print(2**3) # Potenzen
```

```
8
```

```
30 print(pow(2,3)) # Potenzen andere Möglichkeit
```

```
8
```

```
print(14 % 4) # Modulo = der Rest bei der Division
```

```
2
```

```
print(round(13.456))
print(round(15.689))
```

```
13
16
```

```
print(abs(-200))
print(abs(100))
```

```
200
100
```

Versuche es selbst!

Rechnenoperationen:

Erstelle eine neue Python-Datei. Benenne sie so: 3_Rechenoperationen_DeinVorname.py

Schreibe ein kurzes Programm das mehrere Rechenoperationen durchführt, die Ergebnisse in mehreren Variablen speichert und diese Variablen mit dem print() Befehl ausgibt.

6) Benutzereingabe mit input() - User Prompt

Der input() Befehl fordert den/die Benutzer*in auf, eine Eingabe zu machen. Man nennt das einen User Prompt

```
print('Hallo! Mein Name ist Python.')
input('Wie heißt du?')
```

```
Hallo! Mein Name ist Python.
'Christine'
```

Die Eingabe kann in einer Variable gespeichert werden, z.B. um sie später wieder auszugeben.

```
print('Hallo! Mein Name ist Python.')
deinname=input('Wie heißt du?')
print('Schön dich kennenzulernen,')
print(deinname)
```

```
Hallo! Mein Name ist Python.
Schön dich kennenzulernen,
Christine
```

7) Variablen in Ausgabetext

Es gibt 3 praktische Möglichkeiten um Variablen in den Ausgabetext einzubinden:

```
36 print('Schön dich kennenzulernen '+ deinname)
   print('Schön dich kennenzulernen {}'.format(deinname))
   print(f'Schön dich kennenzulernen {deinname}')
```

```
Schön dich kennenzulernen Christine
Schön dich kennenzulernen Christine
Schön dich kennenzulernen Christine
```

Aber Vorsicht wenn du eine Zahl zusammen mit Text ausgeben möchtest:

```
37 alter = 13
   print('Tommy ist ' + alter + ' Jahre alt.')
   # Wie könnte man diesen "Bug" bereinigen?
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)

/var/folders/tt/z9q9vz1x0y907tzlmjs8tvrr0000gn/T/ipykernel_16918/1073161483.py in <module>
      1 alter = 13
----> 2 print('Tommy ist ' + alter + ' Jahre alt.')
      3 # Wie könnte man diesen "Bug" bereinigen?
      4

TypeError: can only concatenate str (not "int") to str
```

Bei den anderen beiden Möglichkeiten wird der Data Type automatisch richtig umgewandelt:

```
38 print('Tommy ist {} Jahre alt.'.format(alter))
   print(f'Tommy ist {alter} Jahre alt.')
```

```
Tommy ist 13 Jahre alt.
Tommy ist 13 Jahre alt.
```

Was ist hier das Problem:

```
39 Kinderzahl = input('Wie viele Kinder haben deine Eltern?')
   Geschwisterzahl = Kinderzahl - 1
   print(f'Du hast also {Geschwisterzahl} Geschwister?')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)

/var/folders/tt/z9q9vz1x0y907tzlmjs8tvrr0000gn/T/ipykernel_16918/339566709.py in <module>
      1 Kinderzahl = input('Wie viele Kinder haben deine Eltern?')
----> 2 Geschwisterzahl = Kinderzahl - 1
      3 print(f'Du hast also {Geschwisterzahl} Geschwister?')
      4

TypeError: unsupported operand type(s) for -: 'str' and 'int'
```

Beim 'Hantieren' mit Variablen musst du auf den Data Type achten. Rechnen kann man nur mit Zahlen, der Befehl input() liest jedoch einen String ein.

Versucht es selbst!

User Eingabe

In Teams zu zweit:

Erstellt eine neue Python-Datei. Benennt sie so: 4_UserEingabe_Vorname1_Vorname2.py

Schreibt ein kurzes Programm, das einen Dialog mit dem/der Benutzer*in führt. Benutzt dafür mindestens drei mal den input() Befehl.

8) Fehlermeldungen

Fehler passieren und können uns beim Lernen helfen. Python sagt dir, wo genau ein Fehler passiert ist.

```
40 print 'Hier gibt es einen Fehler'
```

```
  File "/var/folders/tt/z9q9vz1x0y907tzlmjs8tvrr0000gn/T/ipykernel_16918/24608018.py", line 1
    print 'Hier gibt es einen Fehler'
          ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print('Hier gibt es einen F
```

```
41 print(15/0)
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
/var/folders/tt/z9q9vz1x0y907tzlmjs8tvrr0000gn/T/ipykernel_16918/568293349.py in <module>
----> 1 print(15/0)
      2

ZeroDivisionError: division by zero
```

Wenn du nicht sicher bist was die Fehlermeldung bedeutet, schau im Internet nach. Google Translate hilft dir beim Übersetzen ins Deutsche Es gibt viele Foren wo Programmierer*innen schon einmal dasselbe Problem gepostet haben. Du kannst die Fehlermeldung einfach kopieren und in die Google-Suche einfügen.

Versucht es selbst!

Fehlermeldungen

In Teams zu zweit:

Erstellt eine neue Python-Datei. Benennt sie so:
5_Fehlermeldungen_Vorname1_Vorname2.py

Schaut euch die Datei Fehler.py auf GitHub an und korrigiert sie. Schreibt auf, welche Fehlermeldungen ihr gefunden habt.