

Praktikum Ingenieurinformatik

Teil 1, Programmieren

Termin 3

*Schleifen programmieren,
Funktionen plotten*

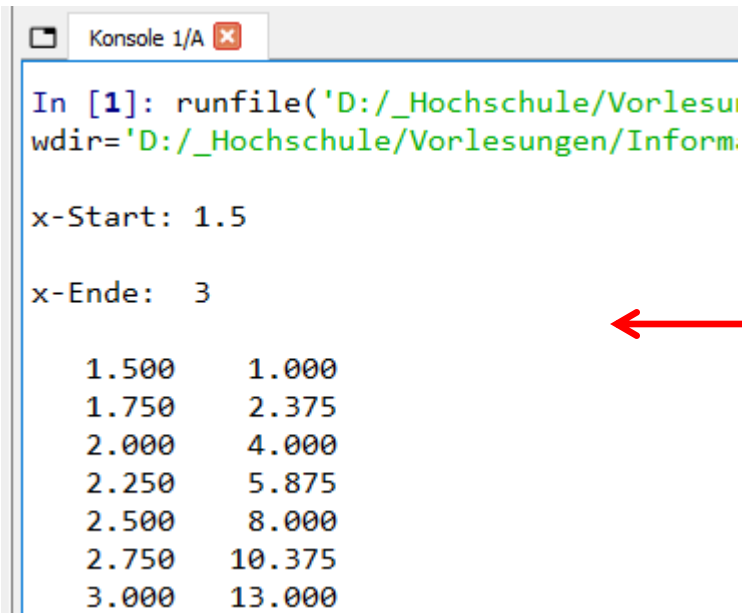
1. Tabellarische Ausgabe von Funktionswerten
2. Funktionen plotten
3. Übungsaufgaben

Aufgabe 1

Das abgebildete Skript berechnet die Funktion $f(x) = 2x - 2$ im Intervall von $-2 \leq x \leq +2$.

- Versuchen Sie, das Skript zu verstehen, geben Sie es in Spyder ein und führen Sie es aus.
- Ändern Sie das Skript so, dass das x-Intervall per Tastatur eingegeben werden kann.
- Alle Werte sollen tabellarisch mit drei Nachkommastellen ausgegeben werden.
- Ändern Sie das Skript so, dass die Funktion $f(x) = 2x^2 - x - 2$ berechnet wird.

```
# Funktionswerte ausgeben
x = -2
while x <= 2:
    y = 2 * x - 2
    print(f"{x} {y}")
    x = x + 0.25
```



```
Konsole 1/A x
In [1]: runfile('D:/_Hochschule/Vorlesu
wdir='D:/_Hochschule/Vorlesungen/Inform

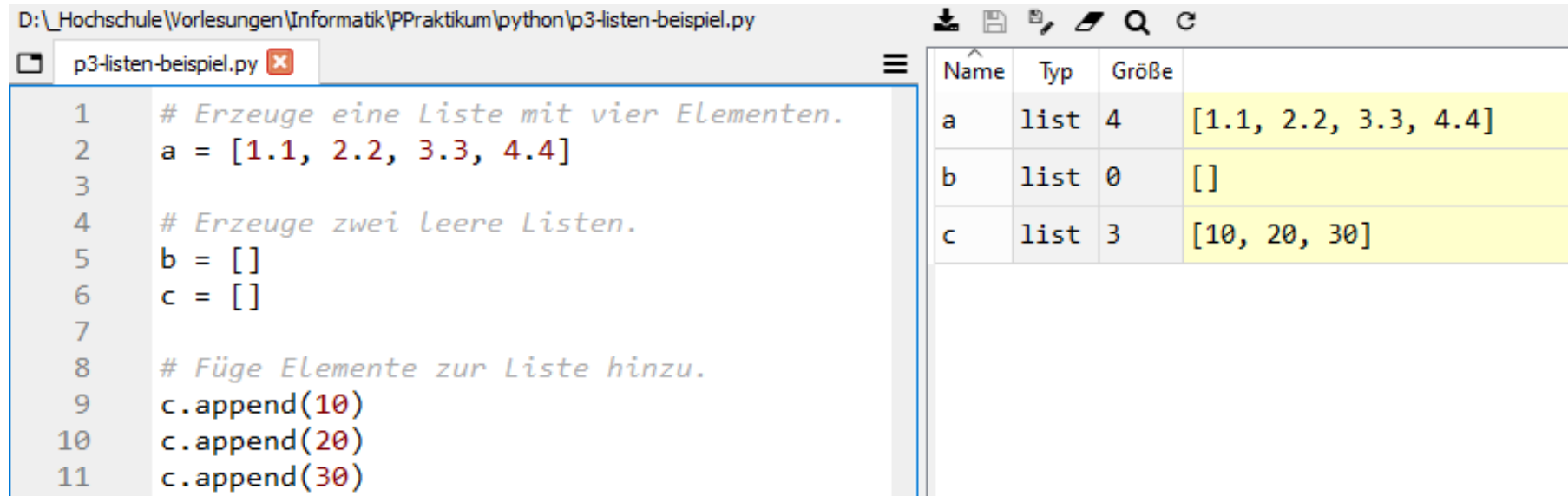
x-Start: 1.5
x-Ende: 3

1.500    1.000
1.750    2.375
2.000    4.000
2.250    5.875
2.500    8.000
2.750    10.375
3.000    13.000
```

1. Tabellarische Ausgabe von Funktionswerten
2. Funktionen plotten
3. Übungsaufgaben

Arbeiten mit Listen

Es kommt häufig vor, dass **Listen von Zahlen** (Messwerte, Berechnungsergebnisse usw.) verarbeitet werden müssen. Zum Glück ist das Arbeiten mit Listen in Python nicht schwer. Die Anzahl der Listen-Elemente ist nur durch den verfügbaren Arbeitsspeicher begrenzt. Versuchen Sie, die folgenden Beispiele zu verstehen:



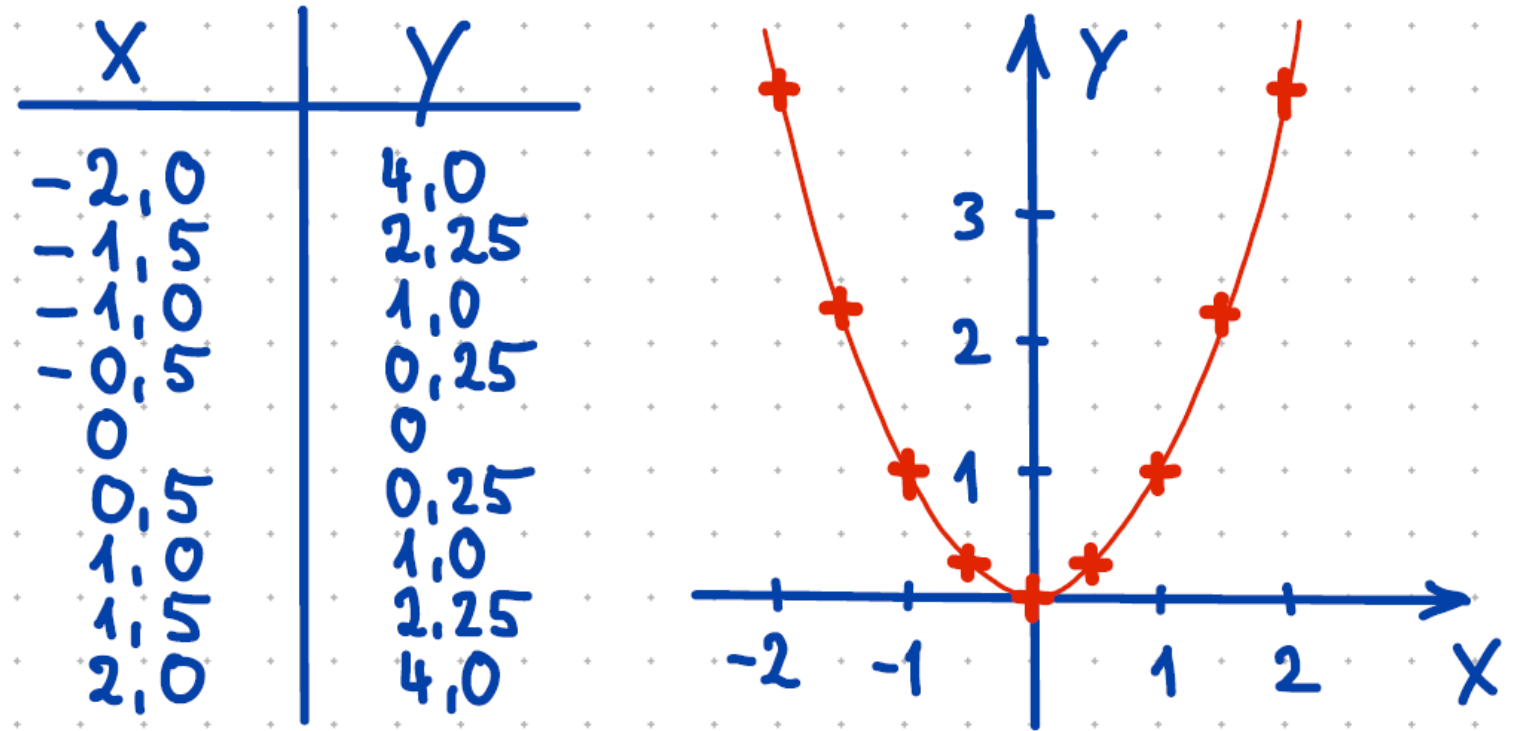
The screenshot shows a Python IDE with a file named `p3-listen-beispiel.py`. The code defines three lists: `a` with four float elements, `b` as an empty list, and `c` as a list with three integers added via `append`. To the right, a variable inspector table displays the state of these variables.

Name	Typ	Größe	
a	list	4	[1.1, 2.2, 3.3, 4.4]
b	list	0	[]
c	list	3	[10, 20, 30]

- Wie fügt man weitere Elemente zu einer Liste hinzu?
- Wie legt man leere Listen an?

Wertetabellen und Funktionsgraphen

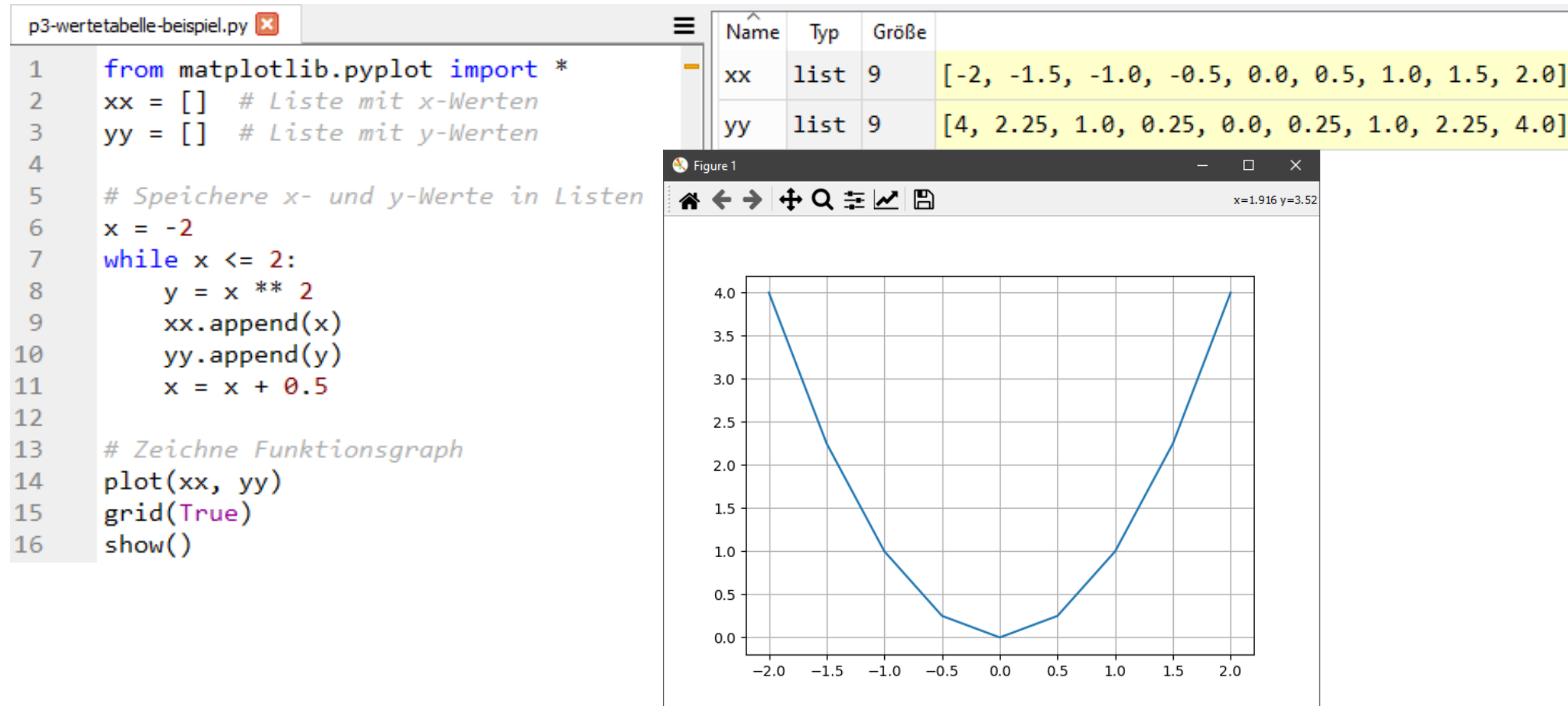
Erinnern Sie sich daran, wie Sie in der Schule **Wertetabellen erstellt und Funktionsgraphen gezeichnet** haben? Das folgende Beispiel zeigt die Wertetabelle und den dazugehörigen Funktionsgraphen einer Normalparabel.



Wertetabellen und Funktionsgraphen in Python

Die linke Spalte der Wertetabelle enthält eine Liste von x-Werten, die rechte Spalte enthält die dazugehörige Liste der y-Werte. Auf dieselbe Art **können auch in Python Wertetabellen erzeugt und Funktionsgraphen gezeichnet** werden.

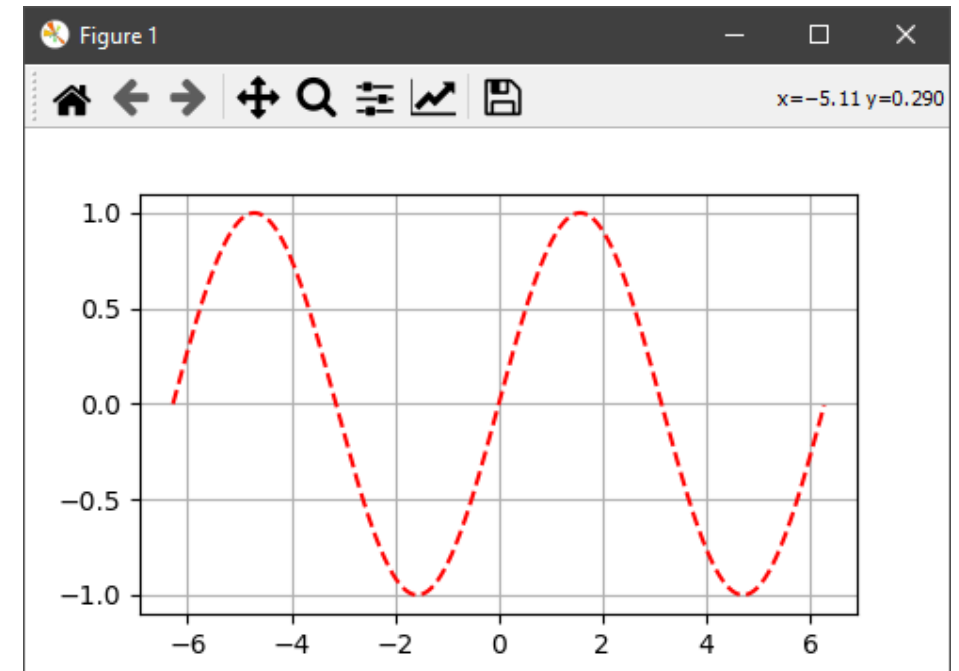
X	Y
-2	4,0
-1,5	2,25
-1	1,0
-0,5	0,25
0	0,0
0,5	0,25
1	1,0
1,5	2,25
2	4,0



Aufgabe 2

Erweitern Sie das Skript aus Aufgabe 1 so, dass zusätzlich zur tabellarischen Ausgabe der Funktionswerte von $f(x) = 2x^2 - x - 2$ auch **der Funktionsgraph gezeichnet** wird.

- Speichern Sie zunächst die berechneten x- und y-Werte in zwei Listen xx und yy.
- Danach erzeugen Sie die grafische Darstellung mit `plot(xx, yy)`, `grid(True)` und `show()`.
- Ersetzen Sie `plot(xx, yy)` durch:
 - `plot(xx, yy, "r-")` – `plot(xx, yy, "r--")`
 - `plot(xx, yy, "g-")` – `plot(xx, yy, "r*")`
 - `plot(xx, yy, "k-")`
- Zeichnen Sie die Sinuskurve $f(x) = \sin(x)$ im Bereich von $-2\pi \leq x \leq +2\pi$. } →
- Wie viele x- und y-Werte werden in Ihrem Skript berechnet? Welche Schrittweite Δx haben Sie gewählt? **Variieren Sie die Schrittweite Δx** und beobachten Sie, wie sich die grafische Darstellung verändert.

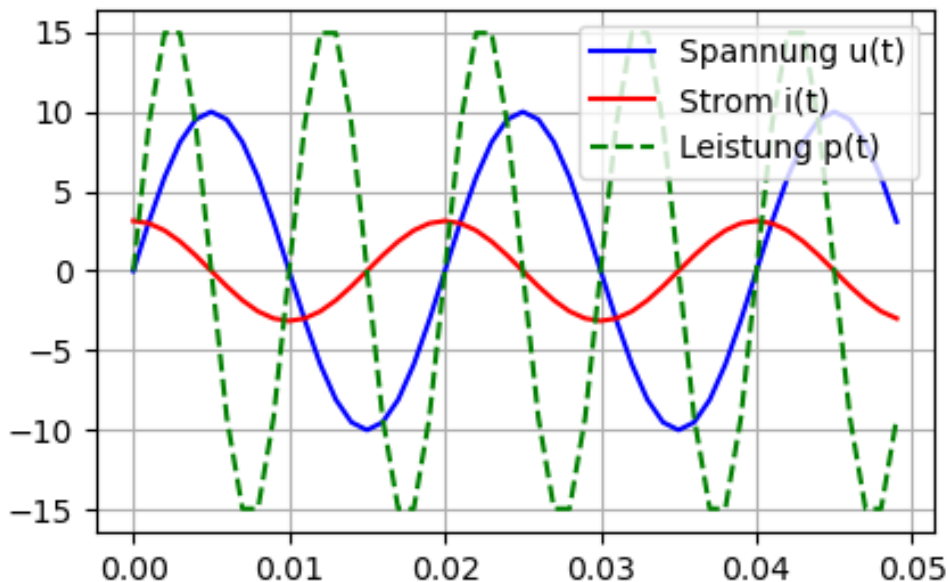
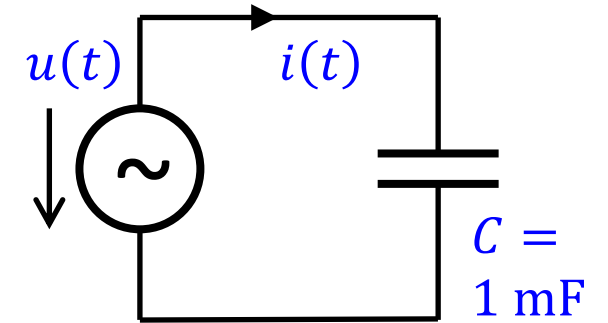


1. Tabellarische Ausgabe von Funktionswerten
2. Funktionen plotten
3. Übungsaufgaben

Übungsaufgabe 1

Ein Kondensator ist an eine Spannungsquelle $u(t) = 10 \cdot \sin(2\pi f \cdot t)$ Volt angeschlossen. Die Frequenz beträgt $f = 50$ Hz, es fließt der Strom $i(t) = \pi \cdot \cos(2\pi f \cdot t)$ Ampere.

- Zeichnen Sie ein Diagramm mit den zeitlichen Verläufen der Spannung $u(t)$, des Stroms $i(t)$ und der Leistung $p(t)$, die vom Kondensator aufgenommen wird (= Produkt aus Spannung und Strom).
- Wie groß ist der zeitliche Mittelwert dieser Leistung?



Tipp:

Möchten Sie mehrere Kurvenverläufe in einem gemeinsamen Diagramm anzeigen (hier: Spannung, Strom, Leistung)? Rufen Sie dazu einfach mehrere `plot()`-Befehle nacheinander auf und anschließend einmalig `show()`, um das Diagramm anzuzeigen.

Übungsaufgabe 2

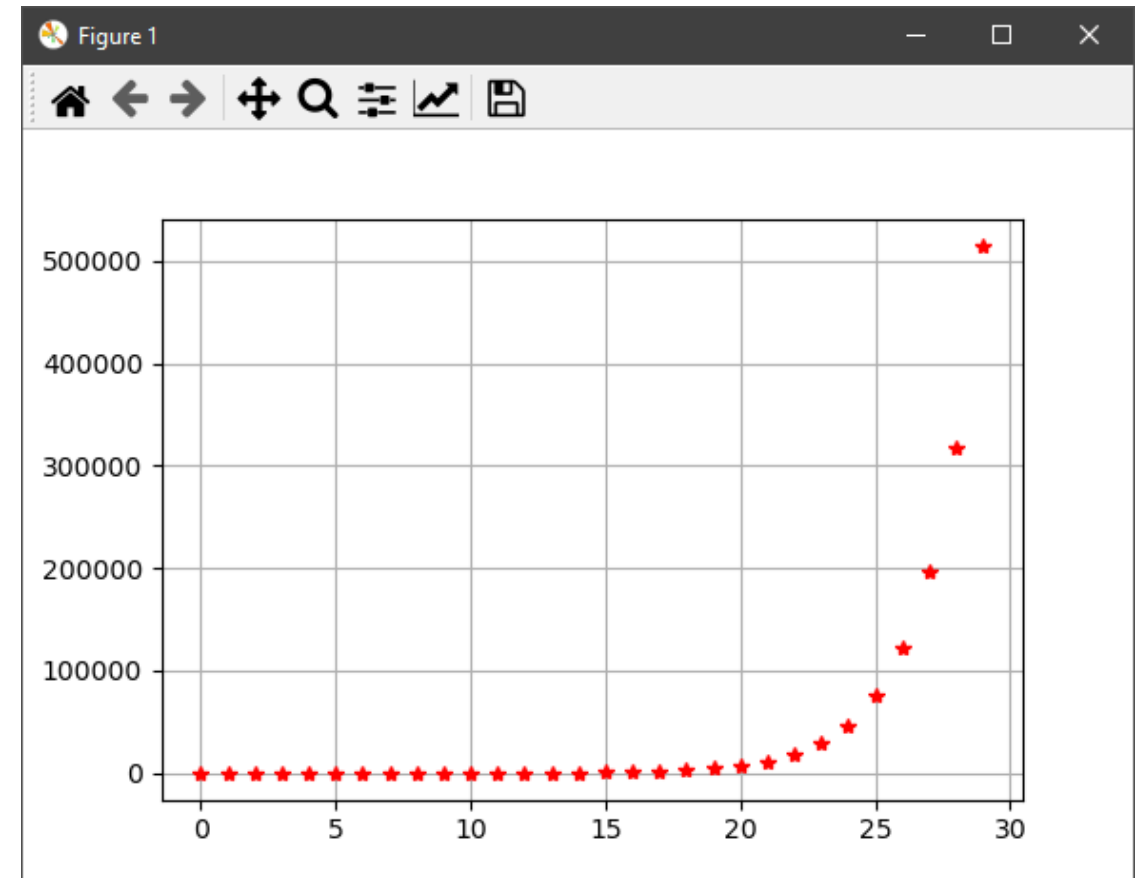
Die Fibonacci-Folge ist die unendliche Folge natürlicher Zahlen, die (ursprünglich) mit zweimal der Zahl 1 beginnt oder (häufig, in moderner Schreibweise) zusätzlich mit einer führenden Zahl 0 versehen ist. Im Anschluss ergibt jeweils die Summe der beiden vorangegangenen Zahlen die unmittelbar danach folgende Zahl:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55 ...

Die darin enthaltenen Zahlen heißen Fibonacci-Zahlen. Benannt ist die Folge nach Leonardo Fibonacci, der damit im Jahr 1202 das Wachstum einer Kaninchenpopulation beschrieb.

Quelle: [1]

Stellen Sie die ersten 30
Fibonacci-Zahlen grafisch dar.



Tipp ...

```
x1 = 0
x2 = 1
x_neu = x1 + x2
print(x1)  # Ausgabe: 0
```

```
x1 = x2
x2 = x_neu
x_neu = x1 + x2
print(x1)  # Ausgabe: 1
```

```
x1 = x2
x2 = x_neu
x_neu = x1 + x2
print(x1)  # Ausgabe: 1
```

```
x1 = x2
x2 = x_neu
x_neu = x1 + x2
print(x1)  # Ausgabe: 2
```

```
x1 = x2
x2 = x_neu
x_neu = x1 + x2
print(x1)  # Ausgabe: 3
```

```
x1 = x2
x2 = x_neu
x_neu = x1 + x2
print(x1)  # Ausgabe: 5
```

... und so weiter – in einer Schleife!

Quellenverzeichnis

- [1] Seite „Fibonacci-Folge“. In: Wikipedia – Die freie Enzyklopädie.
Bearbeitungsstand: 7. August 2022, 10:40 UTC. URL:
<https://de.wikipedia.org/w/index.php?title=Fibonacci-Folge&oldid=225139267>
(Abgerufen: 8. August 2022, 20:56 UTC)