# 20230322_Wirtschaftsinformatik_MV2

March 22, 2023

[1]:
```
# Graphische Darstellung von Daten
```

[2]:
```
# MATPLOTLIB
```

[3]:
```
!pip install matplotlib # wir installieren matplotlib
```

```
Requirement already satisfied: matplotlib in
/Users/h4/anaconda3/lib/python3.9/site-packages (3.5.1)
Requirement already satisfied: cycler>=0.10 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from matplotlib) (1.4.2)
Requirement already satisfied: pillow>=6.2.0 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from matplotlib) (9.0.1)
Requirement already satisfied: packaging>=20.0 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from matplotlib) (21.3)
Requirement already satisfied: numpy>=1.17 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from matplotlib) (1.23.2)
Requirement already satisfied: python-dateutil>=2.7 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from matplotlib) (2.8.2)
Requirement already satisfied: pyparsing>=2.2.1 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from matplotlib) (3.0.4)
Requirement already satisfied: fonttools>=4.22.0 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from matplotlib) (4.25.0)
Requirement already satisfied: six>=1.5 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from python-
dateutil>=2.7->matplotlib) (1.16.0)
```

[5]:
```
import matplotlib.pyplot as plt #wir rufen den Package matplotlib.pyplot auf
```

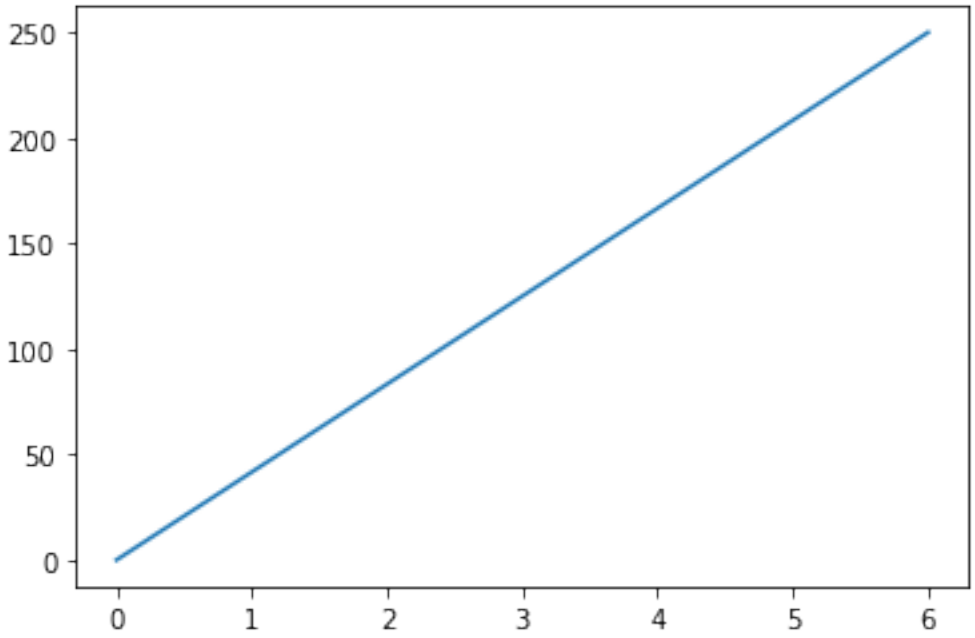
[6]:
```
!pip install numpy
```

```
Requirement already satisfied: numpy in /Users/h4/anaconda3/lib/python3.9/site-
packages (1.23.2)
```

[7]:
```
import numpy as np
```

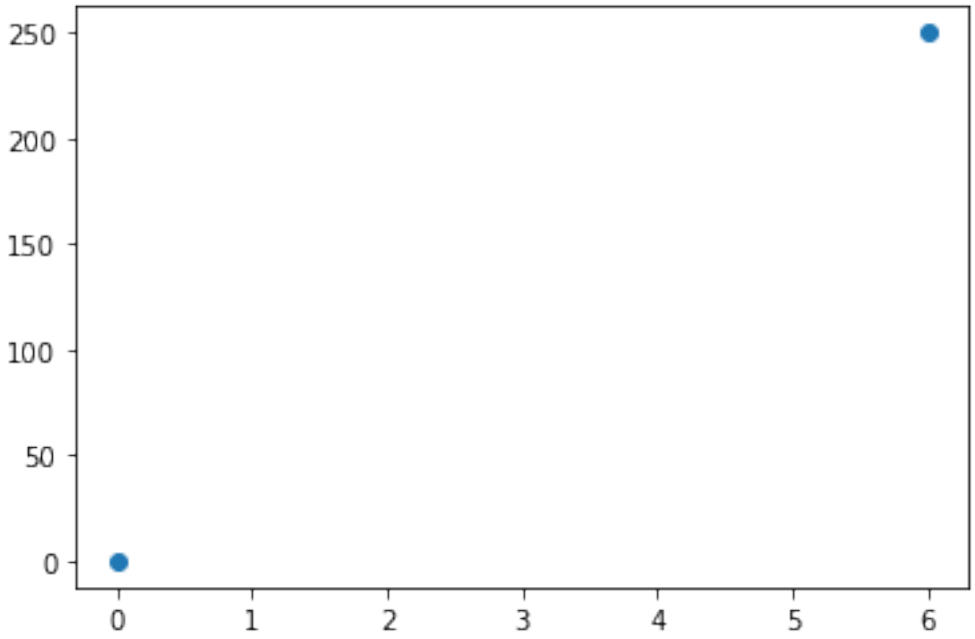
[8]:
```
# beispiel eine linie durch zwei punkte
```

[11]:
```
x_koord = np.array([0,6]) # [] für vektor und () für np.array
y_koord = np.array([0,250])

plt.plot(x_koord, y_koord) # zunächst die x_koordinaten, dann die y_koordinaten
plt.show()
```



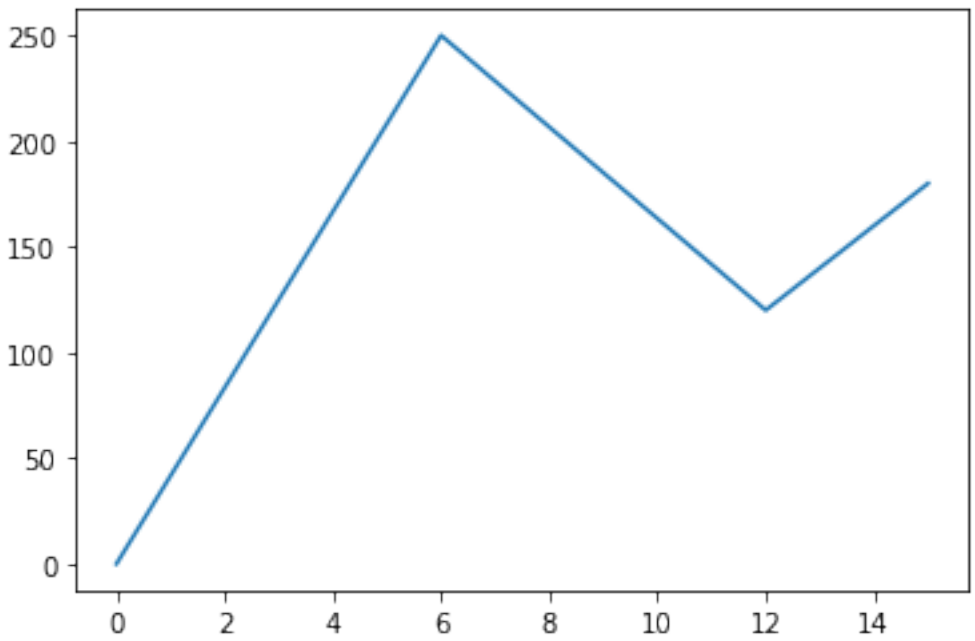

[13]:
```
# beispiel mit zwei Punkte (ohne Linie)
```

[16]:
```
x_koord = np.array([0,6]) # [] für vektor und () für np.array
y_koord = np.array([0,250])

plt.plot(x_koord, y_koord, 'o') # zunächst die x_koordinaten, dann die␣
 ↪y_koordinaten
plt.show()
```

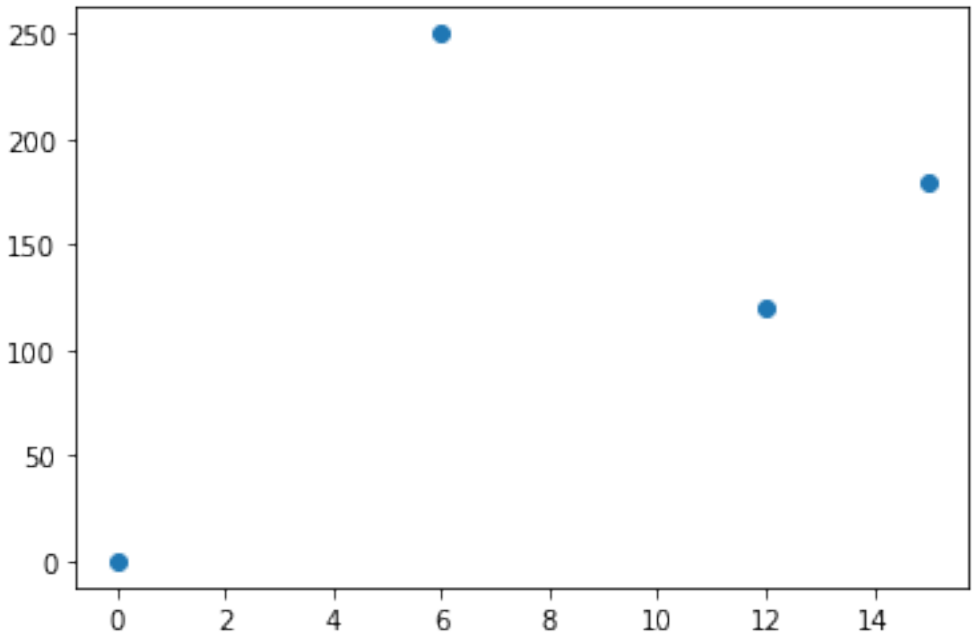



[17]:
```
# beispiel Linie mit mehreren Eckpunkten
```

[21]:
```
x_koord = np.array([0, 6,   12,  15]) # [] für vektor und () für np.array
y_koord = np.array([0, 250, 120, 180])

plt.plot(x_koord, y_koord) # zunächst die x_koordinaten, dann die y_koordinaten
plt.show()
```

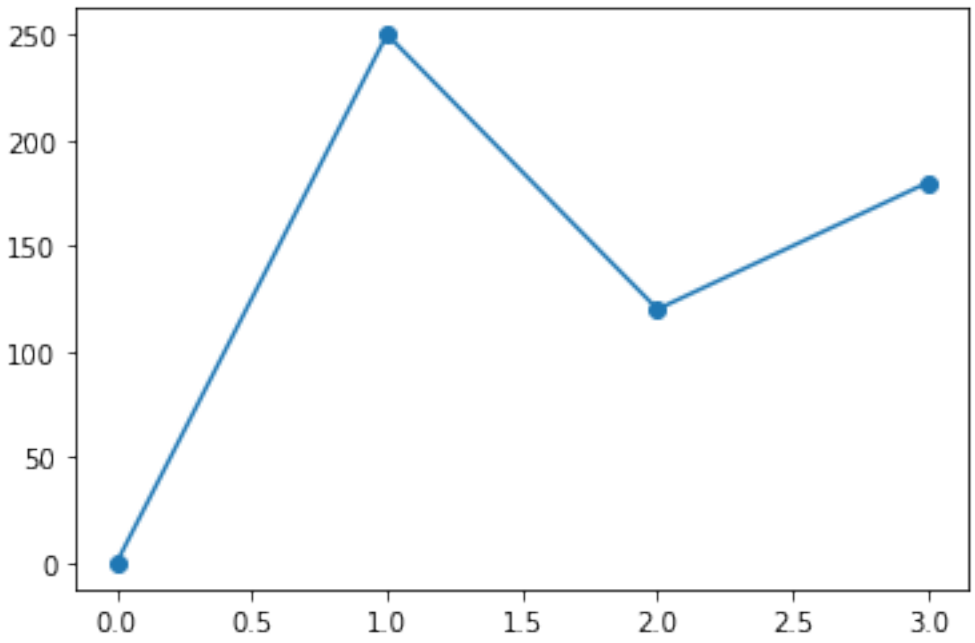


[28]:
```
x_koord = np.array([0, 6,   12,  15]) # [] für vektor und () für np.array
y_koord = np.array([0, 250, 120, 180])

plt.plot(x_koord, y_koord, 'o') # zunächst die x_koordinaten, dann die␣
 ↪y_koordinaten
plt.show()
```



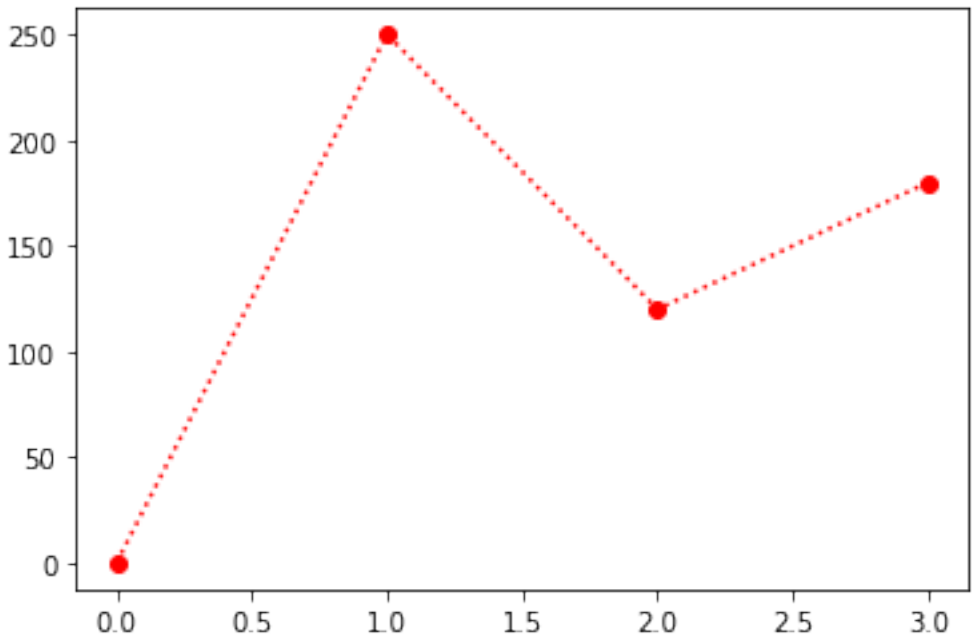

[29]:
```
# beispiel linie+punkte "markers"
```

[50]:
```
y_koord = np.array([0, 250, 120, 180])
plt.plot(y_koord, marker = 'o')
# wir haben hier die x-koordinaten weg gelassen und wird
# per default 0, 1, 2, 3, ... vorausgesetzt.
# gleichzeitig kommt die Linie und den Marker dazu.
plt.show()
```



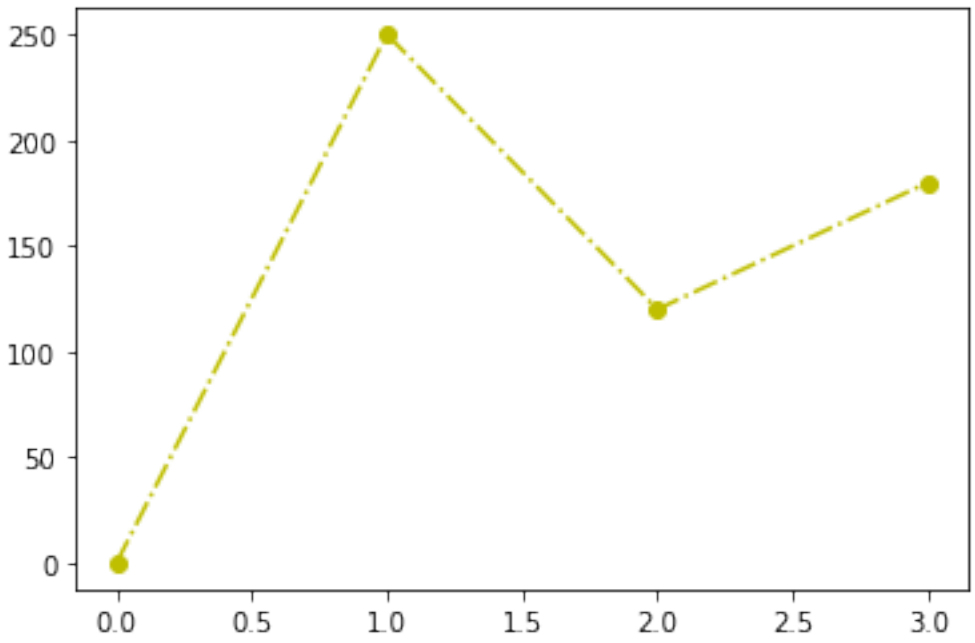

[37]:
```
# beispiel mit einer roten linie druchgestrichen
```

[43]:
```
y_koord = np.array([0, 250, 120, 180])
plt.plot(y_koord, 'o:r')
# rote linie mit markers
# rot kommt von "r"
# die Punkte kommen von "o"
# durchgestrichen kommt von ":"
plt.show()
```



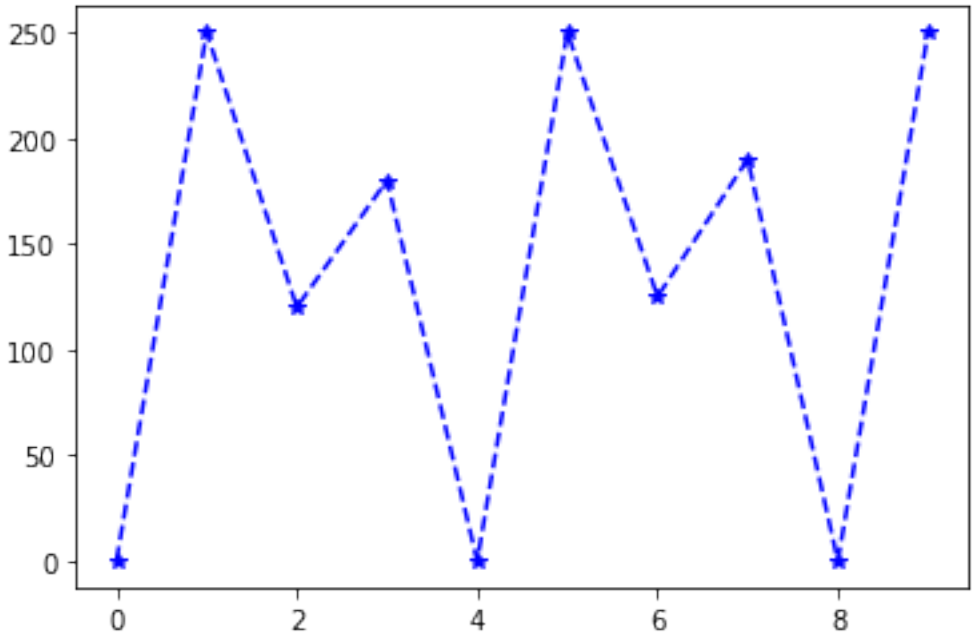

[45]:
```
# beispiel mit einer gelbe linie linie punkte
```

[52]:
```
y_koord = np.array([0, 250, 120, 180])
plt.plot(y_koord, 'o-.y')
# rote linie mit markers
# rot kommt von "r"
# die Punkte kommen von "o"
# linie punkt kommt von "-."
plt.show()
```



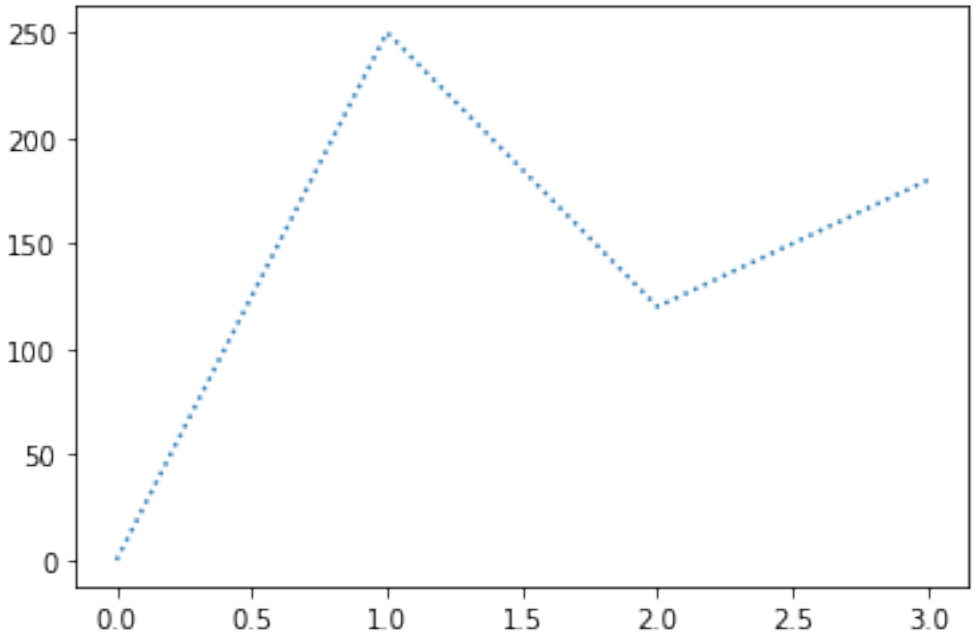

[54]:
```
# beispiel prüfung: bitte stellen Sie eine linie mit farbe blau,
# 10 Punkte, Linie --, marker "*"
```

[57]:
```
y_koord = np.array([0, 250, 120, 180, 0, 250, 125, 190, 0, 250])
# hier werden die Y-Koordinaten von 10 Punkten definiert
plt.plot(y_koord, "*--b")
# hier werden mit "*" die Marker definiert
# hier wird mit "--" die Linie definiert
# hier wird mit "b" die Farbe blau festgelegt
plt.show()
```



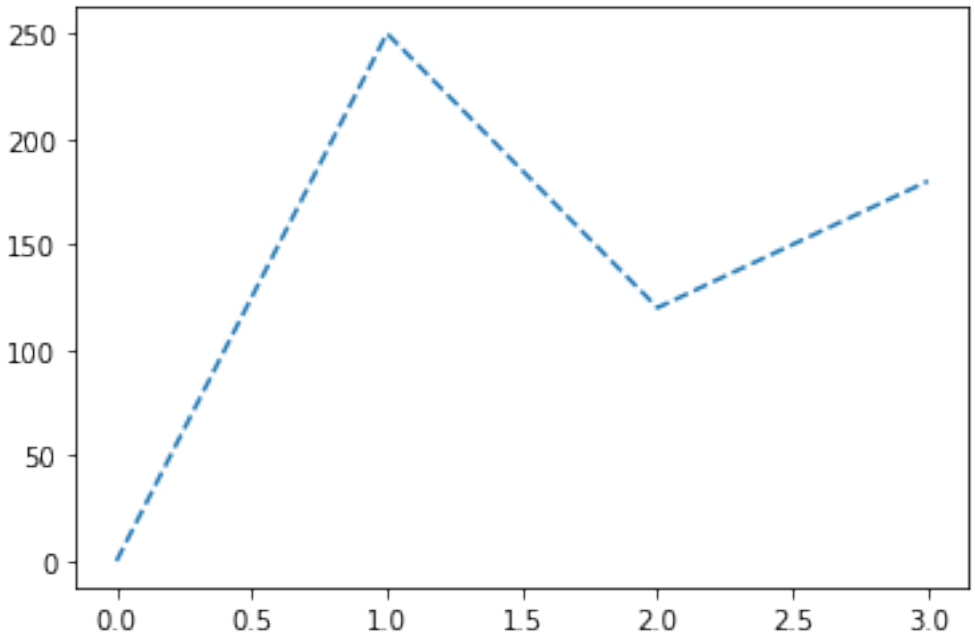

[58]:
```
# beispiel linestyle "dotted"
```

[64]:
```
y_koord = np.array([0, 250, 120, 180])
# hier werden die Y-Koordinaten von 4 Punkten definiert
plt.plot(y_koord, linestyle = 'dotted')
plt.show()
```



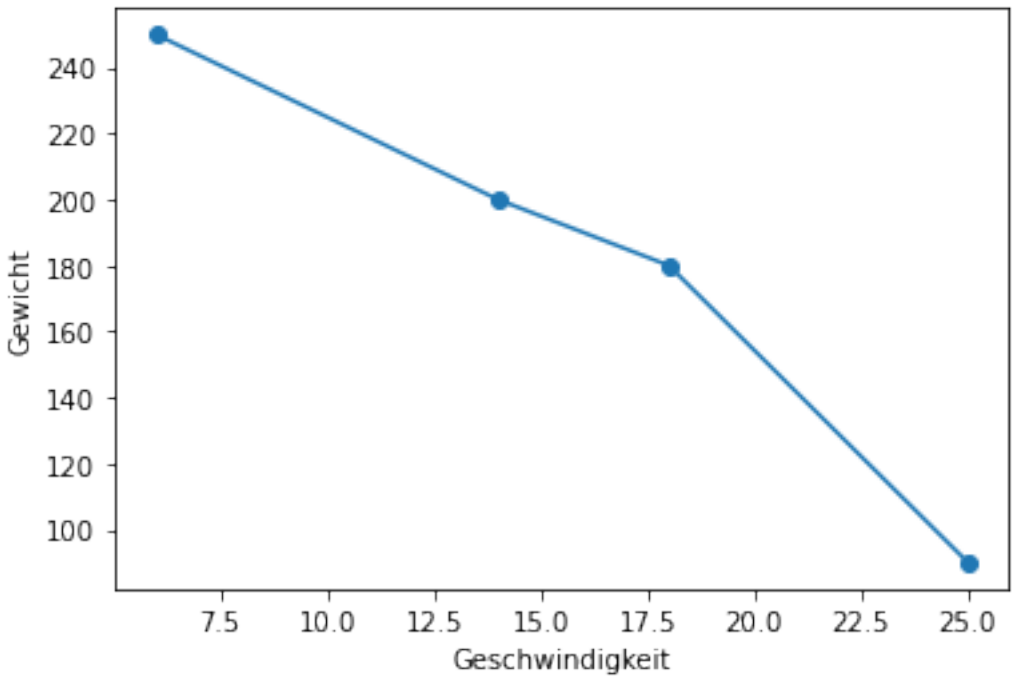
[60]:
```
# beispiel linestyle "dashed"
```

[65]:
```
y_koord = np.array([0, 250, 120, 180])
# hier werden die Y-Koordinaten von 4 Punkten definiert
plt.plot(y_koord, linestyle = 'dashed')
plt.show()
```



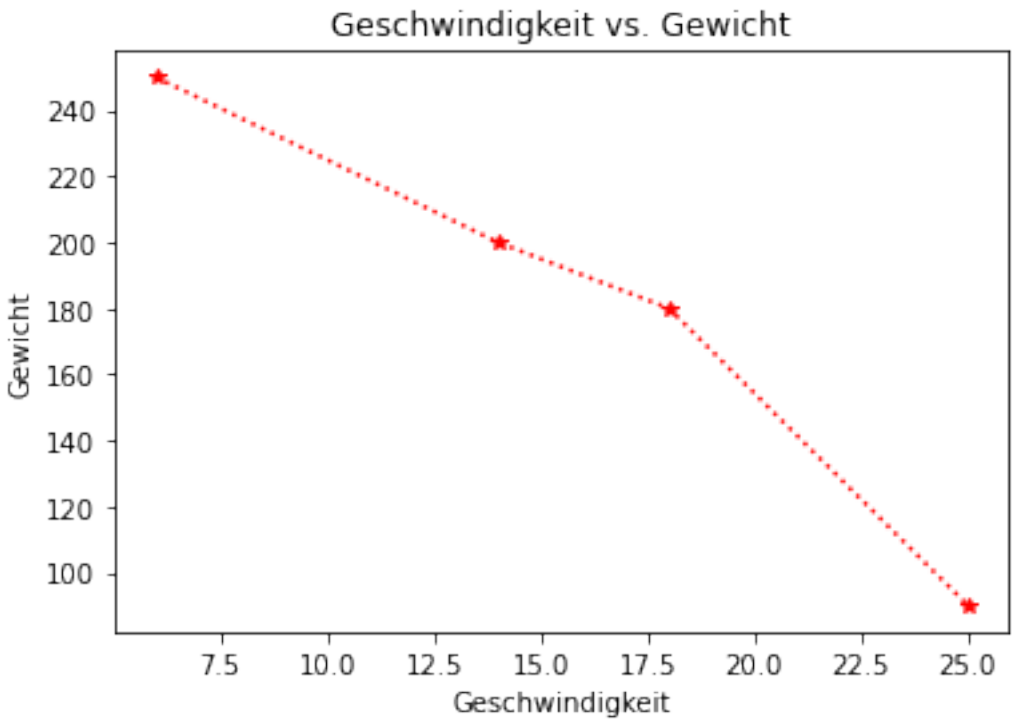

[66]:
```
# beispiel mit labels
```

[72]:
```
x_koord = np.array([6, 14, 18, 25]) # [] für vektor und () für np.array
y_koord = np.array([250, 200, 180, 90])

plt.plot(x_koord, y_koord, marker = 'o') # zunächst die x_koordinaten, dann die␣
 ↪y_koordinaten

plt.xlabel('Geschwindigkeit') # label auf X-Achse
plt.ylabel('Gewicht') # label auf Y-Achse

plt.show()
```




[70]:
```
# beispiel mit Titel
```

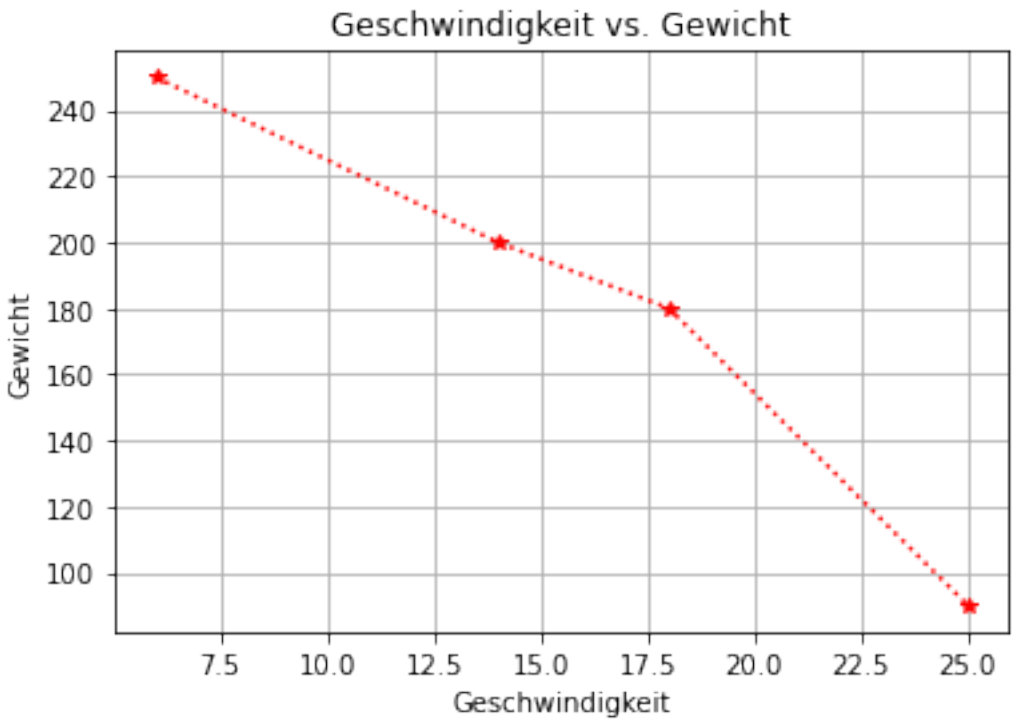
[75]:
```
x_koord = np.array([6, 14, 18, 25]) # [] für vektor und () für np.array
y_koord = np.array([250, 200, 180, 90])

plt.plot(x_koord, y_koord, '*:r') # zunächst die x_koordinaten, dann die␣
 ↪y_koordinaten

plt.title('Geschwindigkeit vs. Gewicht')
plt.xlabel('Geschwindigkeit') # label auf X-Achse
plt.ylabel('Gewicht') # label auf Y-Achse

plt.show()
```




[76]:
```
# beispiel prüfungsfrage: bitte stellen Sie eine Graphik mit Python wie oben␣
↪gezeigt.
```

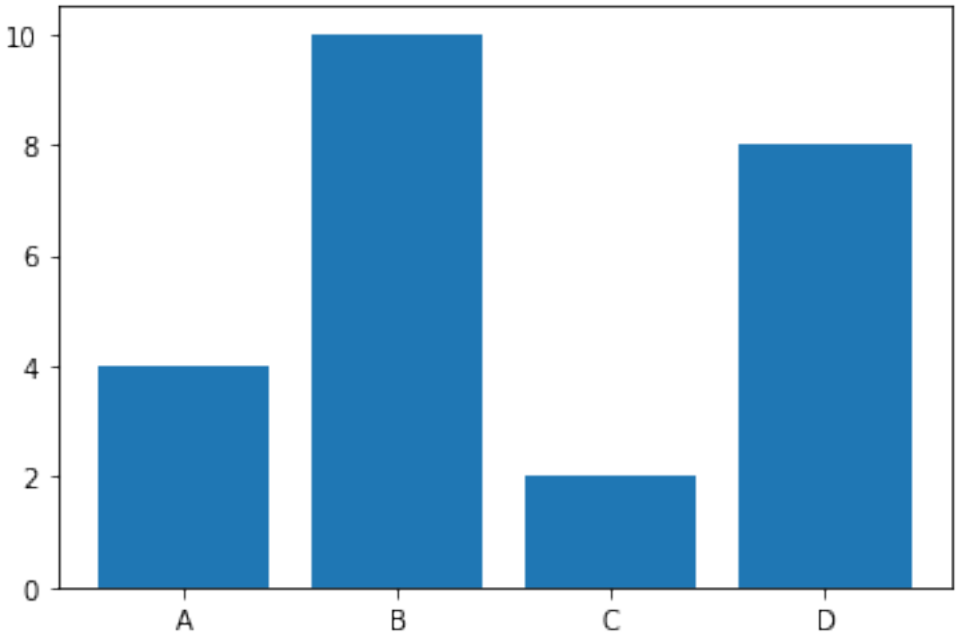
[77]:
```
# beispiel mit "grids" also hilfelinien
```

[79]:
```
x_koord = np.array([6, 14, 18, 25]) # [] für vektor und () für np.array
y_koord = np.array([250, 200, 180, 90])

plt.plot(x_koord, y_koord, '*:r') # zunächst die x_koordinaten, dann die␣
↪y_koordinaten

plt.title('Geschwindigkeit vs. Gewicht')
plt.xlabel('Geschwindigkeit') # label auf X-Achse
plt.ylabel('Gewicht') # label auf Y-Achse

plt.grid() # ein grid wird gezeigt

plt.show()
```

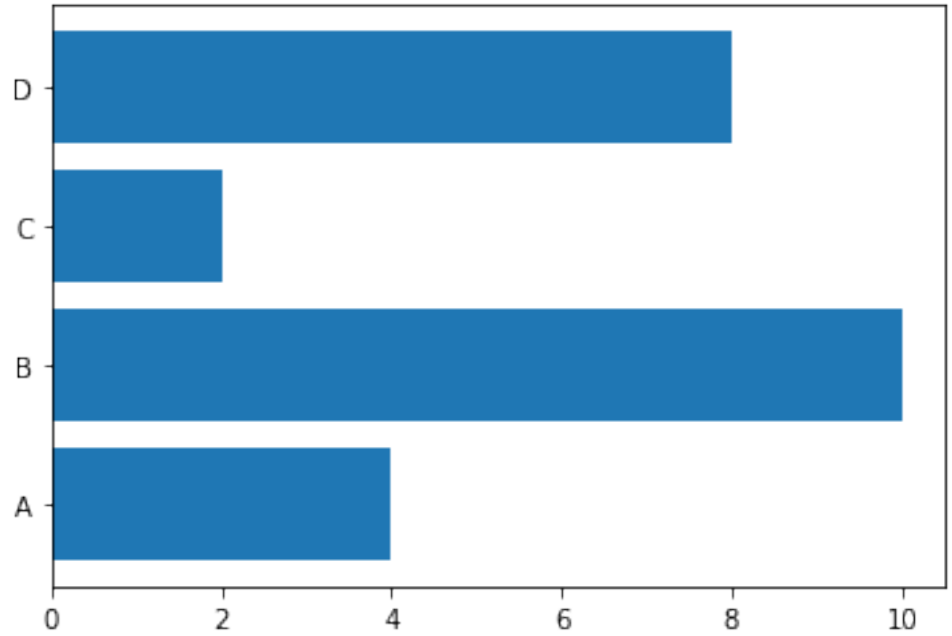



[81]:
```
# beispiel Balkendiagramm
```

[84]:
```
x = np.array(['A', 'B', 'C', 'D']) # Kategorien
y = np.array([4, 10, 2, 8]) # Höhe vom Balken

plt.bar(x, y) # hier wird ein Balkendiagramm vertikal erstellt
plt.show()
```

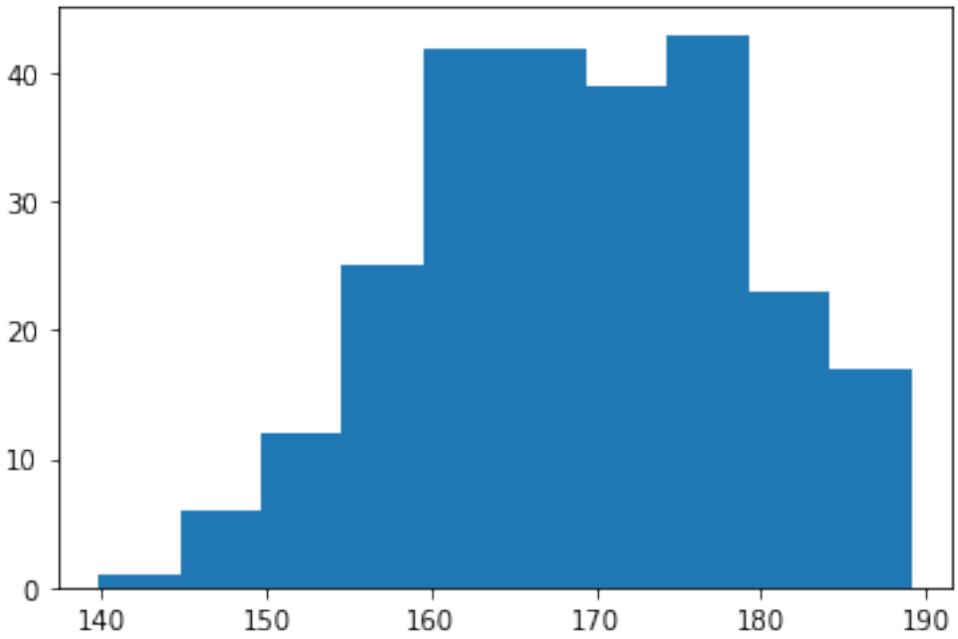



[85]:
```
x = np.array(['A', 'B', 'C', 'D']) # Kategorien
y = np.array([4, 10, 2, 8]) # Höhe vom Balken

plt.barh(x, y) # hier wird ein Balkendiagramm horizontal erstellt
plt.show()
```



[86]:
```
# beispiel histogram
```

[88]:
```
x = np.random.normal(170, 10, 250)
# zufallszahlen in einer Normalverteilung mit Mittelwert 170,
# Std Abweichung 10 und 250 Datensätze

plt.hist(x) # hier wird ein Histogram dargestellt
plt.show()
```

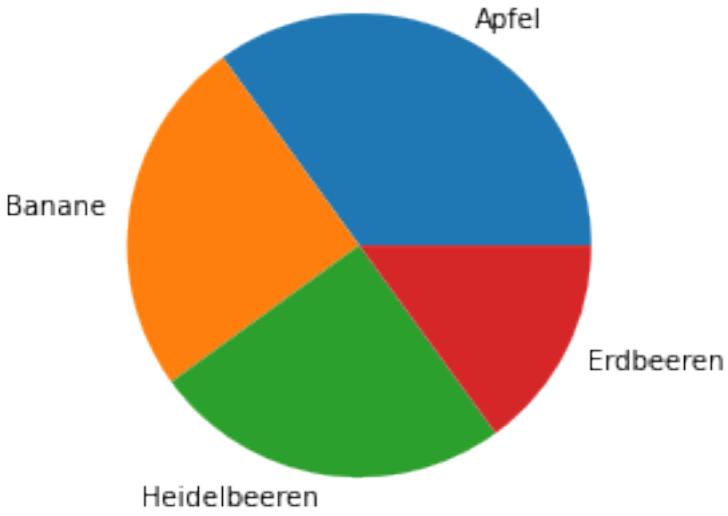



[89]:
```
# beispiel Kuchendiagram
```

[92]:
```
y = np.array([35, 25, 25, 15])

plt.pie(y) # hier wird ein Kuchendiagramm dargestellt
plt.show()
```

[93]:
```
# Kuchendiagramm mit Labels
```

[97]:
```
y = np.array([35, 25, 25, 15])
mylabels = ['Apfel', 'Banane', 'Heidelbeeren', 'Erdbeeren']
plt.pie(y, labels = mylabels) # hier wird ein Kuchendiagramm dargestellt
plt.show()
```



[ ]: