Aufgabe 0

Gegeben ist folgender Array:
`np.array( ["Rot", "Grün", "Rot", "Blau", "Rot", "Grün", "rot"])`
Ermitteln sie die einzigartigen Werte des Arrays mit einer dafür geeigneten `numpy` Funktion.

Aufgabe 0.5

Gegeben sind folgende Arrays:
`array_a = np.array([1, 2, 3, 4, 5])`
`array_b = np.array([3, 4, 5, 6, 7])`
Ermitteln Sie mit `np.isin()` welche Werte von `array_a` in `array_b` vorhanden sind. Benutzen Sie einen boolschen Index um einen Array zu erzeugen der die gemeinsamen Werte enthält.

Aufgabe 0.75

Führen Sie die folgenden Mengenoperationen an den Arrays aus 0.5 durch:

1. Schnittmenge
2. Vereinigungsmenge
3. Differenzmenge

Aufgabe 1

Erstellen Sie ein 2D-Array mit zufälligen Werten und sortieren Sie es einmal nach Zeilen und einmal nach Spalten. Verwenden Sie dafür `np.sort()` mit Angabe des entsprechenden Werts für `axis` .

Aufgabe 2

Gegeben ist folgender Array:
`np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])`

Benutzen Sie `np.sum()` mit dem Argument `axis` um

- die Summe aller Werte für jede Zeile des Arrays zu berechnen.
  Das Ergebnis sollte ein 1D-Array sein, das die Summen der jeweiligen Zeilen enthält.
- die Summe aller Werte für jede Spalte des Arrays zu berechnen.
  Das Ergebnis sollte ebenfalls ein 1D-Array sein, das die Summen der jeweiligen Spalten enthält.

Aufgabe 3

Gegeben ist ein 1D-Array mit Integer-Zahlen, der einige Nullen enthält.
`np.array([10, 0, 5, 20, 0, 25])`
Verwenden Sie `np.where()`, um ein neues Array zu erstellen, in dem die Werte von 100 durch die Elemente des ursprünglichen Arrays geteilt werden. Wenn ein Element des Arrays den Wert 0 hat, soll an dieser Stelle `np.nan` (Not a Number) eingetragen werden.

3.1: Erstellung des Filters

Erstellen Sie einen Filter (in diesem Fall einen booleschen Index), der angibt, welche Elemente im ursprünglichen Array ungleich 0 sind.

3.2: Erstellung des Ergebnis-Arrays

Verwenden Sie den erstellten Filter, um ein neues Array zu erstellen, in dem die Werte von 100 durch die Elemente des ursprünglichen Arrays geteilt werden. An den Stellen, an denen der Filter False ist (also die Elemente 0), sollte `np.nan` eingetragen werden.

3.3 `np.any()` und `np.all()`

Benutzen sie eine der beiden Funktionen um zu testen ob im erstellten Array mindestens ein `np.nan` eingetragen ist. Benutzen Sie eine der Funktionen um zu testen ob im Array kein `np.nan` vorkommt.

3.4: `np.nan` zählen

Ermitteln sie die Anzahl der `np.nan` im Ergebnis. Sie können dazu die Funktion `sum()` in Verbindung mit dem boolschen Index benutzen.