funktionen

October 17, 2025

1 Programmieren einfacher Funktionen

1.1 Kreisfläche

Die Fläche eines Kreises berechnet sich nach der Formel

$$A = \pi r^2$$

wobei A die Fläche π die Kreiszahl - hier mit zwei Nachkommastellen - 3.14 und r den Radius des Kreises darstellt.

Aufgabenstellung: Implementieren Sie eine Funktion `kreisflaeche()` welche ein Argument `r` für Radius entgegennimmt und die Fläche eines Kreises zurückgibt.

[ ]:
```
def kreisflaeche():
    # TODO: implementieren Sie die Funktion
```

1.2 Quotient

Ein Quotient kann unterschiedlich dargestellt werden. Eine Möglichkeit ist die Darstellung als Dezimalzahl.

Aufgabenstellung: Implementieren Sie eine Funktion `quotient()` welche zwei Dezimalzahlen als Argumente entgegennimmt und den Quotienten als Dezimalzahl zurückgibt. Sehen Sie für den Sonderfall der Division durch Null eine Lösung vor.

[ ]:
```
def quotient():
    # TODO: implementieren Sie die Funktion
```

1.3 Fakultät

Die Fakultät ist jene Funktion, welche das Produkt der natürlcihen Zahlen bis n berechnet und wird als $n!$ geschrieben. Es gilt also

$$n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$$

Wobei zu beachten ist, dass $0! = 1$ gilt.

Aufgabe: Implementieren Sie eine Funktion `fakultaet` welche als Argument $n \in \mathbb{N}$ entgegennimt.

[ ]:
```python
def fakultaet():
    # TODO: implementieren Sie die Funktion
```

1.4 Quadratische Gleichungen (Mitternachtsformel)

Quadratische Gleichungen der Form $ax^2 + bx + c = 0$ können mit der Formel

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - ac}}{2a}$$

gelöst werden.
Für die Implementierung ergibt sich daraus allerdings das Problem, dass die Formel zwei Resultate ergibt und eine Python Funktion nur ein Objekt zurückgeben kann.

Als Workaround können die beiden Resultate in ein Tupel gepackt werden. Ein Tupel ist eine unveränderliche Zusammenstellung mehrer Werte. In Python werden sie in Klammern dargestellt. Die Zuweisung eines Tupels zu einer Variabel sieht dementsprechend folgendermassen aus:

`t = (wert_1, wert_2, ..., wert_n)`

Die detaillierten eigenschaften von Tupeln können hier ausser Acht gelassen werden. Es reicht x_1 und x_2 einem Tupel `resultat` zuzuweisen und `resultat` zurückzugeben.

Aufgabe: Implementieren Sie eine Funktion `mitternachtsformel`, welche die Argumente a, b und c entgegennimt und das Tupel x_1 und x_2 zurückgibt.

[ ]:
```python
def mitternachtsformel():
    # TODO: implementieren Sie die Funktion
```

1.5 Musterlösungen

1.5.1 Kreisfläche

[ ]:
```python
def kreisflaeche(r: float) -> float:
    """
    Berechnet die Fläche eines Kreises anhand des gegebenen Radius.

    Parameter
    ----------
    r : float
        Der Radius des Kreises.

    Rückgabewert
    ------------
    float
        Die Fläche des Kreises.
    """
    PI = 3.14  # Näherungswert für die Kreiszahl Pi
    a = PI * r ** 2  # Formel: Fläche = Pi * r^2
    return a
```

1.5.2 Quotient

[ ]:
```python
def quotient(z: float, n: float) -> float:
    """
    Berechnet den Quotienten zweier Zahlen.

    Gibt eine Fehlermeldung zurück, falls der Nenner null ist.

    Parameter
    ----------
    z : float
        Der Zähler der Division.
    n : float
        Der Nenner der Division.

    Rückgabewert
    ------------
    float oder str
        Der berechnete Quotient oder eine Fehlermeldung als Zeichenkette,
        falls eine Division durch Null versucht wurde.
    """
    fehlermeldung = "Eine Division durch Null ist unzulässig."  # Fehlermeldung␣
↪für Division durch Null
    if n == 0:
        return fehlermeldung  # Fehlerfall: Division durch Null
    else:
        q = z / n  # Berechnung des Quotienten
        return q
```

1.5.3 Fakultät

[ ]:
```python
def fakultaet(n: int) -> int:
    """
    Berechnet die Fakultät einer nicht-negativen ganzen Zahl n.

    Die Fakultät n! ist das Produkt aller positiven ganzen Zahlen
    von 1 bis n. Für n < 0 ist die Fakultät nicht definiert.

    Parameter:
        n (int): Die Zahl, deren Fakultät berechnet werden soll.

    Rückgabewert:
        int: Die Fakultät von n, oder eine Fehlermeldung als String bei n < 0.
    """
    if n < 0:
        # Fakultät ist für negative Zahlen nicht definiert
        return f'Die Fakultät für {n} ist nicht definiert.'
```

```
    elif n == 0:
        # Per Definition ist 0! = 1
        return 1
    else:
        prod = 1  # Initialisiere das Produkt mit 1
        for i in range(1, n + 1):
            prod *= i  # Multipliziere prod mit dem aktuellen Wert von i
        return prod  # Gib das berechnete Produkt zurück
```

1.5.4 Mitternachtsformel

[ ]:
```
def mitternachtsformel(a: float, b: float, c: float) -> tuple:
    """
    Löst eine quadratische Gleichung der Form ax² + bx + c = 0 mit der␣
↪Mitternachtsformel.

    Parameter
    ----------
    a : float
        Der Koeffizient vor x² (muss ungleich 0 sein).
    b : float
        Der Koeffizient vor x.
    c : float
        Der konstante Term.

    Rückgabewert
    ------------
    tuple oder str
        Ein Tupel mit den beiden Lösungen (x1, x2) oder eine Fehlermeldung als␣
↪Zeichenkette,
        falls die Gleichung keine reellen Lösungen besitzt.
    """
    # Berechnung der Diskriminante: d = b² - 4ac
    d = b ** 2 - 4 * a * c
    if d < 0:
        return "Diese Gleichung hat keine Lösung."  # Keine reellen Lösungen␣
↪vorhanden
    else:
        # Berechnung der beiden Lösungen mit der Mitternachtsformel:
        # x1 = (-b + d ** (1/2)) / (2a)
        # x2 = (-b -  d ** (1/2)) / (2a)
        x1 = (-b + d ** (1/2)) / (2 * a)
        x2 = (-b - d ** (1/2)) / (2 * a)
        return (x1, x2)
```