

Eine Funktion fasst eine Rechnung unter einem **Namen** zusammen. Sie bekommt Werte (**Parameter**) und gibt mit `return` ein **Ergebnis** zurück:

```
def brutto(netto):  
    return netto + netto * 0.19  
  
preis = brutto(80)  
print(preis)           # 95.2
```

 **Aktivierung Jede/r für sich:** **(a)** Was liefert `brutto(200)`? **(b)** Berechnen Sie mit *zwei* Aufrufen den Bruttobetrag einer Rechnung über 80 € und 20 € netto. **(c)** Welche Zeilen gehören zur Funktion — woran erkennen Sie das?

Brick-Flow berechnet Rechnungen immer gleich: **Anzahl mal Einzelpreis**. Statt das jedes Mal neu zu tippen, schreiben wir eine Funktion mit **zwei Parametern**:

```
def rechnungsbetrag(anzahl, einzelpreis):  
    # Ergänzen Sie: Ergebnis berechnen und zurueckgeben  
  
print(rechnungsbetrag(12, 80))    # erwartet: 960  
print(rechnungsbetrag(3, 80))    # erwartet: 240
```



Aktivierung Zu zweit: Ergänzen Sie den Funktionsrumpf (eine Zeile mit **return**) und prüfen Sie beide Aufrufe.

Ab 10 Paletten gibt es 5 % Rabatt (wie in Übung 2). Jetzt soll das eine Funktion erledigen — und dabei die Funktion aus Aufgabe 1 **wiederverwenden**:

```
def endbetrag(anzahl, einzelpreis):  
    summe = rechnungsbetrag(anzahl, einzelpreis)  
    # ab 10 Paletten -> summe * 0.95 zurueckgeben,  
    # sonst -> summe zurueckgeben  
  
print(endbetrag(12, 80))  
print(endbetrag(8, 80))
```



Aktivierung Zu zweit: Ergänzen Sie den `if/else`-Block mit je einem `return`. Was geben die beiden Aufrufe aus?

Normalerweise ist der Versand **ab 50 €** frei, sonst kostet er **4,95 €**. Premiumkunden bekommen eine niedrigere Grenze. Ein Parameter mit **Standardwert** (`grenze=50`) muss beim Aufruf nur angegeben werden, wenn er abweicht:

```
def versandkosten(bestellwert, grenze=50):  
    # Ergänzen Sie: 0.0 ab der Grenze, sonst 4.95  
  
print(versandkosten(42))           # Normalkunde  
print(versandkosten(42, 40))       # Premiumkunde, Grenze 40
```



Aktivierung Zu zweit: Ergänzen Sie die Funktion. Welche Werte erwarten Sie für die beiden Aufrufe — *bevor* Sie ausführen?

Ein Kollege hat eine Rabattfunktion geschrieben (10 % Rabatt). Danach soll eine Versandpauschale von **50€** aufgeschlagen werden:

```
def preis_mit_rabatt(preis):  
    print(preis * 0.9)  
  
endpreis = preis_mit_rabatt(2000) + 50  
print(endpreis)
```

 **Aktivierung Zu zweit:** **(a)** Was erscheint beim Ausführen — und warum kommt trotzdem ein Fehler? **(b)** Ändern Sie *ein* Wort so, dass **1850.0** herauskommt.

Kein Code als Vorlage mehr — schreiben Sie die Funktionen selbst (**def**, Parameter, **return**, ggf. **if/elif/else**).

Aufgabe 5 — Deckungsbeitrag. Schreiben Sie eine Funktion `deckungsbeitrag(preis, variable_kosten, menge)`, die (Preis – variable Kosten) × Menge zurückgibt. Testen Sie mit **80 €** Preis, **52 €** variablen Kosten und **250** Paletten.

Aufgabe 6 — Kundenstatus. Schreiben Sie eine Funktion `kundenstatus(umsatz)`: unter **1.000 €** → "Bronze", unter **5.000 €** → "Silber", sonst "Gold". Testen Sie mit **800**, **3200** und **7500**.



Aktivierung Zu zweit: Lösen Sie beide Aufgaben im Notebook und vergleichen Sie Ihre Ergebnisse.

Aufgabe 7 — Break-even-Menge. Ab welcher Menge deckt Brick-Flow die Fixkosten?
Die Break-even-Menge ist

$$\text{Menge} = \frac{\text{Fixkosten}}{\text{Preis} - \text{variable Kosten je Stück}}$$

Schreiben Sie `break_even(fixkosten, preis, variable_kosten)` und testen Sie mit **14.000 €** Fixkosten, **80 €** Preis und **52 €** variablen Kosten je Palette.

Zusatz: Was ändert sich, wenn die Fixkosten auf **21.000 €** steigen? Sie müssen dafür nur *einen* Aufruf ergänzen — nicht die Funktion ändern.



Aktivierung Zu zweit: Schätzen Sie zuerst das Ergebnis — dann rechnen Sie es in Python nach.

Aufgabe 8 — Pflaster für einen Lagerhof (für Wirtschaftsingenieur:innen). Ein Hof ist **15 m** lang und **8 m** breit. Pro m^2 braucht man **36** Pflastersteine, dazu **5 %** Verschnitt. Ein Stein kostet **0,50 €**; **ab 2.000 €** Materialwert gibt es **5 % Rabatt**.

- a)** `flaeche(laenge, breite)` — Fläche in m^2
- b)** `steinbedarf(flaeche, steine_pro_m2=36)` — Steine inkl. Verschnitt
- c)** `materialkosten(steine, preis_je_stein=0.5)` — Kosten mit/ohne Rabatt



Aktivierung Zu zweit: Schreiben Sie die drei Funktionen und verketteten Sie die Aufrufe.

Geschafft

Eigene **Funktionen** mit Parametern, **return** und Standardwerten, Funktionen **kombiniert**, **print** vs. **return** — Rechenwege einmal schreiben, beliebig oft nutzen.