Codeklasse

Die Zaubermaschine

Thema: Eigene Funktionen & Parameter

Name: _____

Klasse: _____ Datum: _____

Grundlagen & Merkkasten

WISSEN

Bisher wurden Befehle Zeile für Zeile nacheinander ausgeführt. Müssen ähnliche Schritte mehrfach ausgeführt werden, führt wiederholtes Kopieren von Code schnell zu unübersichtlichen Programmen.
Mit eigenen Funktionen definierst du wiederverwendbare Bausteine. Über Parameter übergibst du veränderliche Werte an die Funktion, sodass derselbe Codeblock flexibel mit unterschiedlichen Eingabedaten arbeiten kann.

`def sag_hallo(name):` `  print('Hallo', name)`	Definiert eine Funktion mit dem Namen 'sag_hallo' und dem Parameter 'name'.
`sag_hallo('Alex')`	Ruft die Funktion auf und übergibt das Argument 'Alex'.

`def duplizieren(x):`
`  ergebnis = x * 2`
`  print(ergebnis)`
	Führt Rechenoperationen innerhalb des gekapselten Funktionsblocks aus.

1. Code-Puzzle (Reihenfolge & Einrückung)

AUFGABE 1

Bringe die Code-Blöcke in die richtige Reihenfolge und achte auf die korrekte Einrückung, um eine Funktion `transformieren` zu definieren und anschließend für `Sternenstaub` aufzurufen.

[ ]	`def transformieren(zutat, menge):`
[ ]	`print('Transformiere ' + str(menge) + 'x ' + zutat + '...')`
[ ]	`ausbeute = menge * 3`
[ ]	`print('Ergebnis: ' + str(ausbeute) + ' Einheiten erzeugt!')`
[ ]	`transformieren('Sternenstaub', 6)`
[ ]	`function transformieren(zutat, menge):`
[ ]	`transformieren(6, 'Sternenstaub')`

Hinweis: Trage in die eckigen Klammern die korrekte Reihenfolge (1, 2, 3...) ein. Streiche überflüssige oder fehlerhafte Zeilen (Distraktoren) durch!

2. Fehlersuche & Code-Reparatur

AUFGABE 2

Im folgenden Skript soll eine Funktion `braue_trank` definiert und zweimal mit unterschiedlichen Parametern aufgerufen werden. Es haben sich jedoch 2 Fehler eingeschlichen:
1. Im Funktionskopf fehlt das Schlüsselwort zur Funktionsdefinition.
2. Beim zweiten Aufruf wurde der Parametername statt des gewünschten Werts `4` übergeben.

Korrigiere den Code, sodass beide Trank-Mischungen fehlerfrei berechnet und ausgegeben werden.

Fehlerhafter Ausgangs-Code:

```
1  braue_trank(name, stufe):
2      energie = stufe * 10
3      print("Trank:", name, "| Energie:", energie)
4
5  braue_trank("Heiltrank", 2)
6  braue_trank("Zaubertrank", stufe)
```

Deine Korrektur / Notizen:

3. Programmier-Challenge

AUFGABE 3

Programmiere eine eigene Zaubermaschinen-Funktion zur Veredelung von Materialien:

1. Definiere eine Funktion namens `vervielfachen` mit den beiden Parametern `material` und `menge`.
2. Innerhalb der Funktion:
- Berechne in einer Variablen `ergebnis` das Produkt aus `menge` und `3`.
- Gib folgenden Text auf der Konsole aus: `Ergebnis: <ergebnis> x <material>`
3. Rufe die Funktion am Ende außerhalb der Definition auf mit:
- `material` = `'Sternenstaub'`
- `menge` = `6`

Dein Python-Code:

Transfer-Aufgabe: Das DRY-Prinzip und Modularisierung

AUFGABE 4

In der Softwareentwicklung gilt die Grundregel DRY („Don't Repeat Yourself“ – Wiederhole dich nicht).
Erkläre anhand eines Beispiels, warum es vorteilhafter ist, wiederkehrende Befehlsfolgen in eine Funktion mit Parametern auszulagern, anstatt denselben Code mehrfach an verschiedenen Stellen im Programm einzufügen. Gehe dabei besonders auf Fehlersuche und Änderungen der Programmlogik ein.