Name: Punkte: /15P Note:

1) Schleifen und Kontrollstrukturen 3P

```python
sales = [15, 3, 0, 22, 14, 0, 1]

streak = 0
score = 0

for sale in sales:
    if sale > 14:
        streak += 1
        score += 2
    elif sale > 3:
        score += 1
    else:
        streak = 0

    print(f"{sale} {streak} {score}")
```

Welche Ausgabe wird gemacht?

Bitte wenden!

Notenschlüssel: 14-15: Sehr gut, 12-13: Gut, 10-11: Befriedigend, 8-9: Genügend, 0-7 Nicht Genügend

2) Objektorientierte Programmierung 3P

```
class Employee:
    count = 0
    total_salary = 0

    def __init__(self, name, base):
        self.name = name
        self.base = base
        self.salary = 0
        Employee.count += 1

    def pay(self, hours=1):
        salary = self.base * hours
        Employee.total_salary += salary
        self.salary += salary

class Manager(Employee):
    def __init__(self, name, base, bonus=10):
        super().__init__(name, base)
        self.bonus = bonus

    def pay(self, hours=1):
        salary = (self.base + self.bonus) * hours
        Employee.total_salary += salary
        self.salary += salary

e1 = Employee("Alice", 20)
e2 = Manager("Charlie", 20)
e3 = Employee("Bob", 25)
```

| Befehl | `Employee.count` | `Employee.total_salary` | `e1.salary` | `e2.salary` | `e3.salary` |
|---|---|---|---|---|---|
| Initialisierung | | | | | |
| `e1.pay(5)` | | | | | |
| `e2.pay(hours=2)` | | | | | |
| `e3.pay()` | | | | | |

3) Dictionary 3P

| Anweisung | x | y | z |
|---|---|---|---|
| `info = {"x": 2, "y": 4}` | 2 | 4 | |
| `info["z"] = info["x"] * info["y"]` | | | |
| `info["x"] += info["z"]` | | | |
| `info["y"] = info.get("x", 11)` | | | |
| `info["z"] -= info["y"]` | | | |
| `info["x"] = info["x"] // 3` | | | |
| `del info["x"]` | | | |
| `info["y"] = len(info)` | | | |
| `info["z"] = sum(info.values())` | | | |

4) Datenbearbeitung

```
names = ["Anna", "Bob", "Carla", "Anton"]
filtered = [n for n in names if n[0] == "A"]

mapped = {}

for name in filtered:
    vowels = 0
    for letter in name:
        if letter.lower() in 'aeiou':
            vowels += 1
    mapped[name] = vowels
```

Wert für filtered:

Wert für mapped:

Notenschlüssel: 14-15: Sehr gut, 12-13: Gut, 10-11: Befriedigend, 8-9: Genügend, 0-7 Nicht Genügend

5) List Comprehension 3P

| **values=** | **result=** | **result** |
|---|---|---|
| `[5, -3, 9, -1, 0, 7]` | `[v if v >= 0 else 0 for v in values]` | |
| `range(1, 10)` | `[x * x for x in values if x % 2 == 0]` | |
| | `[len(w) for w in values]` | `[3, 2, 4]` |
| `[3, 11, 7, 20, 15]` | | `[-3, -11, -7, -20, -15]` |
| `{"name": "Bob", "age": 20, "city": "Graz"}` | `[k for k in values.keys()]` | |
| `["tree", "sun", "sky", "mountain", "house"]` | | `["TREE", "MOUNTAIN", "HOUSE"]` |
| `range(3)` | `[f"Item {i}" for i in values]` | |