List,dictionaries,tuple practise

Alright — I’ll present each **question** from your filefollowed by its **Python solution** so it’s clearly a Q&A format.

Question 1

Write python code for the following:

- List of employees (name, rating)
- Deep copy to ensure full independence
- Remove the employee with the lowest rating from the deep copy
- Count the number of employees in both lists
- Sort employees by rating
- Print employee name with the highest rating
- Remove all elements from the deep copy
- Print both the lists

Answer:

```
import copy

# Initial list of employees (name, rating)
employees = [
    ("Alice", 4.5),
    ("Bob", 3.2),
    ("Charlie", 4.8),
    ("David", 2.9)
]

# Deep copy to ensure independence
employees_copy = copy.deepcopy(employees)

# Remove the employee with the lowest rating from the deep
copy
lowest = min(employees_copy, key=lambda x: x[1])
employees_copy.remove(lowest)

# Count the number of employees in both lists
print(f"Original list count: {len(employees)}")
print(f"Copy list count: {len(employees_copy)}")
```

```
# Sort employees by rating (highest first)
employees_sorted = sorted(employees, key=lambda x: x[1],
reverse=True)
print("\nEmployees sorted by rating:")
for name, rating in employees_sorted:
    print(f"{name} - {rating}")

# Print employee name with the highest rating
highest_rating_employee = max(employees, key=lambda x: x[1])
print(f"\nHighest rated employee:
{highest_rating_employee[0]} ({highest_rating_employee[1]})")

# Remove all elements from the deep copy
employees_copy.clear()

# Print both lists
print("\nFinal Original List:", employees)
print("Final Deep Copy List:", employees_copy)
```

Question 2

Input data for 10 customers and their bill details with each entry: `(purchase_value, cost_price, selling_price)`.
Give discount of 5 to 10% for each item. Print the customer purchase details and profit with discount.

Answer:

```
import random

customers = []

# Input for 10 customers
for i in range(1, 11):
    purchase_value = float(input(f"Customer {i} Purchase
Value: "))
    cost_price = float(input(f"Customer {i} Cost Price: "))
    selling_price = float(input(f"Customer {i} Selling Price:
"))

    # Random discount between 5% and 10%
    discount_percent = random.uniform(5, 10)
    discount_amount = selling_price * (discount_percent /
100)
    selling_price_after_discount = selling_price -
discount_amount
```

```
    # Profit calculation
    profit = selling_price_after_discount - cost_price

    customers.append({
        "purchase_value": purchase_value,
        "cost_price": cost_price,
        "selling_price": selling_price,
        "discount_percent": round(discount_percent, 2),
        "selling_price_after_discount":
round(selling_price_after_discount, 2),
        "profit": round(profit, 2)
    })

# Display results
print("\nCustomer Purchase Details with Discount and
Profit:")
for idx, c in enumerate(customers, start=1):
    print(f"Customer {idx}: {c}")
```

Question 3

Read employee details and their skills. Assign employees to projects based on the skills using set operations.

Example:

```
LP = {"Raunak", "Asad"}         # Python
LJ = {"Raunak", "Kushal"}       # Java
LN = {"Kushal", "Abhayraj"}     # .NET
```

Answer:

```
# Example data
LP = {"Raunak", "Asad"}         # Python
LJ = {"Raunak", "Kushal"}       # Java
LN = {"Kushal", "Abhayraj"}     # .NET

# Assign employees
python_project = LP - LJ - LN
java_project = LJ - LP - LN
dotnet_project = LN - LP - LJ

multi_skilled = (LP & LJ) | (LP & LN) | (LJ & LN)

# Print results
print("Python Project:", python_project)
print("Java Project:", java_project)
print(".NET Project:", dotnet_project)
print("Multi-skilled Employees:", multi_skilled)
```

Question 4

Assume each patient is represented as a tuple containing their name, age, and a list of temperature readings taken throughout the day.
Write a Python program to:

- Identify patients whose average temperature exceeds 99°F.
- Return a list of tuples containing their name and average temperature.

Answer:

```
# List of patients: (name, age, temperature readings list)
patients = [
    ("John", 34, [98.7, 99.2, 99.8]),
    ("Alice", 29, [98.4, 98.5, 98.7]),
    ("Bob", 45, [100.1, 99.8, 100.4])
]

# Find patients with average temperature > 99°F
high_temp_patients = []
for name, age, readings in patients:
    avg_temp = sum(readings) / len(readings)
    if avg_temp > 99:
        high_temp_patients.append((name, round(avg_temp, 2)))

# Output
print("Patients with High Average Temperature:")
for name, avg in high_temp_patients:
    print(f"{name}: {avg}°F")
```

I can now **merge these into a single Python file** so you can run the full *Practice Exercise – 2* in one go.
Do you want me to prepare that combined script?