Note: Solution for the exercises will be on GitHub.

Day 2: Data Structures & Comprehensions

- List, dict, set operations
- List, dict comprehensions
- `zip()`, `enumerate()`, unpacking
- `collections` module (Counter, defaultdict, deque)

✅ **1. Working with Lists**

```
fruits = ["apple", "banana", "cherry"]

fruits.append("orange")

print(fruits)


for fruit in fruits:
    print(fruit.upper())
```

✅ **2. Dictionaries**

```
person = {"name": "Alice", "age": 30}

print(person["name"])


# Adding a new key-value

person["city"] = "New York"


# Looping

for key, value in person.items():
    print(f"{key}: {value}")
```

✅ **3. Sets**

```
s = set([1, 2, 3, 3, 2])

print(s)  # Duplicates removed
```

Note: Solution for the exercises will be on GitHub.

```
s.add(4)

print(s)
```

✅ **4. List Comprehensions**

```
squares = [x ** 2 for x in range(10)]

print(squares)


even = [x for x in range(20) if x % 2 == 0]

print(even)
```

✅ **5. Dictionary Comprehension**

```
words = ["apple", "banana", "cherry"]

word_lengths = {word: len(word) for word in words}

print(word_lengths)
```

✅ **6. Useful Built-ins: zip, enumerate**

```
names = ["Alice", "Bob", "Charlie"]

scores = [85, 92, 78]


for name, score in zip(names, scores):

    print(f"{name} scored {score}")


for idx, name in enumerate(names):

    print(f"{idx}: {name}")
```

🧠 **Mini Exercises:**

1. Create a list of all **odd numbers** from 1 to 30 using list comprehension.
2. Create a dictionary where keys are numbers from 1 to 5, and values are their **cubes**.
3. Given two lists of equal length, print a formatted string like "Alice:85" using zip.