# Quiz 06-26: Lists, Sets, and Dictionaries

## Part A — Lists (create, index, update)

### 1) Index + update

Fill in the blanks:

```
colors[2] = "purple"
print(colors)
```

Output:

```
['red', 'green', 'purple', 'yellow']
```

### 2) List + loop (short answer)

1. What is `nums[-1]` ? **5**
2. What is `len(nums)` ? **5**
3. After `nums.append(9)` , the new last item is **9**

### 3) What does it print?

Output:

```
[10, 20, 30, 40]
[10, 20, 30, 40]
```

### 4) Build a new list

One correct fill:

```
squares.append(x * x)
```

Output:

```
[1, 4, 9, 16]
```

## Part B — Loops (nested for + while)

### 5) Nested loop: count prints

It runs **12** times.

### 6) Nested loop: coordinate pairs

Fill in the blanks:

- first blank: **2**
- second blank: **2**

### 7) While loop: stop at a sentinel

Fill in the blanks:

- sentinel: **0**
- step: **1**

Output:

```
14
```

### 8) Fix the bug (nested loop)

Correct line:

```
pairs.append((n, ch))
```

### 9) What does it print? (while loop)

Output:

```
32
```

## Part C — Sets (membership + operations)

### 10) Duplicates disappear

Output:

```
5
2
```

### 11) Membership: list vs set

Faster choice (for large data): **set**

Reason:

```
A set membership check is O(1) on average.
A list membership check is O(n) on average.
```

### 12) Set operations: friends (order may vary)

1. Mutual friends (intersection): `{"Mia", "Kai"}`
2. All friends (union): `{"Tom", "Mia", "Zoe", "Kai", "Leo"}`
3. Alice but not Bob (difference): `{"Tom", "Zoe"}`

### 13) Fix the code: empty set

Correct code:

```
seen = set()
```

## Part D — Dictionaries (create + traversal + membership)

### 14) What does it print? (keys only)

Output:

```
True
False
```

## 15) Safe counting pattern

Fill in the blanks:

- first blank: **0**
- second blank: **1**

Output (order may vary):

```
{'B': 1, 'A': 3, 'N': 2}
```

## 16) Traverse `.items()`

Fill in the blank:

```
items()
```

So the loop is:

```
for k, v in pets.items():
    print(k, "->", v)
```

## 17) Find the biggest value

Fill in the blanks:

```
for name, score in scores.items():
    if score > best_score:
        best_score = score
        best_name = name
```

Output:

```
Ben
```

## 18) Bug hunt: wrong key check

Fix the `if` line:

```
if "age" in person:
```

# Part E — Word frequency (dict + loops + sets)

## 19) Word frequency

Fill in the blanks:

- first blank: **0**
- second blank: **1**

Output (order may vary):

```
{'a': 3, 'b': 2, 'c': 1}
```

## 20) Unique words + frequency (short answer)

1. How many unique words are there? **4**
2. Best data type for storing unique words: **set**

## 21) Combine list + set + dict

Fill in the blanks:

```
unique_animals = set(animals)

# ...
freq[a] = 0
freq[a] = freq[a] + 1
```

Output:

```
3
{'cat': 3, 'dog': 2, 'bird': 1}
```

(order of the dictionary may vary)

# Part F — Time Complexity (choose: O(1), O(n), O($n^2$))

## 22) Complexity: single loop

**O(n)**

## 23) Complexity: nested loops

**O(n²)**

---

## 24) Complexity: membership in a list

**O(n)**

---

## 25) Complexity: membership in a set

**O(1)** (average case, in this quiz)

---

## 26) Choosing the faster approach

1. Idea A: **O(n)**
2. Idea B: **O(n)** (build is O(n); each later query is O(1))
3. Better for many different queries: **Idea B**

---

## 27) List vs set membership

1. If `bad` is a **list**: **O(n²)**
2. If `bad` is a **set**: **O(n)**

---

## 28) Remove duplicates but keep order

Fill in the blanks:

```
seen = set()

# ...
if x not in seen:
    unique.append(x)
    seen.add(x)
```

Output:

```
[1, 2, 3, 4]
```

---

## 29) Nested loops + dictionary

Fill in the blank:

```
r * c
```

Output:

```
6
```