Quiz 08–18: Loops and Time Complexity

Part A — Core ideas (concepts)

1) While-loop facts (True/False)

1. True
2. True
3. True

---

2) “In-place” (fill in the blanks)

- In-place means we **do** change the same list.
- We **don't** create a new list.

---

3) Two pointers (fill in the blanks)

- `left` starts at **0**
- `right` starts at **`len(s) - 1`**
- We keep looping while **`left < right`**

---

4) Digits with `%` and `//` (fill in the blanks)

- `n % 10` gives the **last** digit of `n`.
- `n // 10` **removes** the last digit of `n`.

---

5) Negative indices (fill in the blanks)

- `letters[-1]` is **"e"**
- `letters[-4]` is **"b"**

---

6) Quick time complexity (choose one)

Time complexity: **O(1)**

Part B — What does it print?

7) Running total with a stop

Output: `text 2 5 9 done`

8) Stop at the first multiple of 4

Output: `text 3 6 7 stop`

9) Search with a Boolean `found`

Output: `text False`

10) While-loop doubling

Output: `text 1 2 4 8`

11) Digits trace

Output: `text 2 0 4`

12) Nested loops: build short codes

Output: `text ['X1', 'X2', 'Y1', 'Y2']`

13) Move zeros (scan step only)

Output: `text [2, 1, 0, 1] write = 2`

Part C — Fill / fix / write code

14) Running total (fill in the blanks)

One correct fill: ```python numbers = [1, 2, 3, 4]

total = 0 for x in numbers: total = total + x

print("Sum =", total) ```

---

15) Filter (fill in the blanks)

One correct fill: ```python numbers = [5, 12, 9, 15, 10]

big = [] for x in numbers: if x > 10: big.append(x)

print(big) ```

---

16) Fix the indentation

Corrected code: `python for _ in range(3): print("hi")`

---

17) Fix the infinite loop (add ONE line)

Add one line to change `n` each time: ```python n = 3

while n >= 0: print(n) n = n - 1 ```

---

18) In-place reverse (fill in the blanks)

```
items = ["A", "B", "C", "D"]

left = 0
right = len(items) - 1

while left < right:
    temp = items[left]
    items[left] = items[right]
    items[right] = temp

    left = left + 1
    right = right - 1

print(items)
```

19) Debug move-zeros (2 bugs)

- Line A should stop before `read == n`.
- Line B should only move `write` when a non-zero is copied.

Correct lines: - Line A: `while read < n:` - Line B: move the increment inside the `if` block, like this:

```
numbers = [0, 1, 0, 3, 0, 2]

read = 0
write = 0
n = len(numbers)

while read < n:
    if numbers[read] != 0:
        numbers[write] = numbers[read]
        write = write + 1   # moved inside the if
    read = read + 1
```

20) Write code: first even index (use `break`)

One correct solution: ```python numbers = [7, 9, 5, 12, 3, 8]

idx = -1 for i in range(len(numbers)): if numbers[i] % 2 == 0: idx = i break

print(idx) ```

Part D — Digits and number problems

21) By hand: sum of digits

Answer: **10**

22) Write code: count digits (while-loop)

One correct solution: ```python n = 87531

count = 0 while n > 0: count = count + 1 n = n // 10

print(count) ```

23) Write code: print factors

One correct solution: ```python n = 20

for i in range(1, n + 1): if n % i == 0: print(i) ```

24) Prime check (what does it print?)

Output: `text False`

Part E — Two pointers and swapping

25) Two pointers: print first mismatch (write code)

This string has **no mismatch**, so it should print `No mismatch`.

One correct solution: ```python s = "ABCDXDCBA"

left = 0 right = len(s) - 1 found = False

while left < right: if s[left] != s[right]: print(s[left], s[right]) found = True break left = left + 1 right = right - 1

if not found: print("No mismatch") ```

26) Swap first and last item only (write code)

One correct solution: ```python letters = ["p", "y", "t", "h", "o", "n"]

temp = letters[0] letters[0] = letters[-1] letters[-1] = temp

print(letters) ```

Part F — Time complexity (best/worst)

27) Search with `break`

- Best case time complexity: **O(1)**
- Worst case time complexity: **O(n)**

---

28) Nested loops with early stop

- Best case time complexity: **O(1)** (the first pair works)
- Worst case time complexity: **$O(n^2)$**