Worksheet 16: Move Zeros Using Two Pointers

Name: _____ Date: _____

Instructions

- Answer in the blanks.
- For “write code” questions, write valid Python code.
- For “what is the output” questions, write the exact output (including spaces/newlines).

Part A — Key ideas

1) Fill in the blanks: what does “in-place” mean?

Choose from `do` and `don't`.

In-place means we _____ change the input list itself. We _____ create a new list.

2) Two pointers: `read` and `write`

Fill in the blanks.

- `read` starts at _____ (`0` or `1`?) and moves to the _____ (`left` or `right`?) end.
- `write` starts at _____ (`0` or `1`?) and points to the next place to write a _____ (`zero` or `nonzero`?) number.
- If `numbers[read]` is zero, we only move _____ (`read` or `write`?).
- If `numbers[read]` is nonzero, we copy it to `numbers[_____]` (`read` or `write`?).

Part B — Trace the algorithm

3) What is the output? (scan only)

What does this code print?

```
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
    read = read + 1

print(numbers)
print("write =", write)
```

Output:

```

```

4) After scan, fill zeros

Suppose after the scan step we have:

```
numbers = [1, 3, 2, 3, 0, 2]
write = 3
n = 6
```

Now run:

```
for k in range(write, n):
    numbers[k] = 0
```

What is `numbers` after filling zeros?

Answer: `numbers =` _____

5) What is the output? (full algorithm)

What does this code print?

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

write = 0
n = len(numbers)

for read in range(n):
    if numbers[read] != 0:
        numbers[write] = numbers[read]
        write = write + 1

for k in range(write, n):
    numbers[k] = 0

print(numbers)
```

Output:

```

```

Part C — Fix and write code

6) Debugging (2 bugs)

This code is trying to move zeros to the end, but it has **two bugs** (Line A and B).

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

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

while read <= n:          # Line A
    if numbers[read] != 0:
        numbers[write] = numbers[read]
    write = write + 1     # Line B
    read = read + 1
```

Corrected lines:

- Line A: `while _____:`
- Line B: `_____`

7) Fill in the missing code (all while-loops)

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

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

# Step 1: move all nonzeros to the front (keep order)
while read < n:
    if numbers[read] != 0:
        numbers[write] = numbers[read]
        write = write + 1
    read = read + 1

# Step 2: fill zeros from write to the end
k = _____________________
while _____________________:
    numbers[k] = _____________________
    k = _____________________

print(numbers)
```

Expected output:

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