Sheet 3: Tail Recursion Based MCQs (5 Questions)

Focus: Recursion where the **recursive call is the last operation**, enabling optimization and no extra post-call work.

MCQ 1: Tail Sum

```
function tailSum(n, acc):
    if n == 0:
        return acc
    return tailSum(n - 1, acc + n)

print(tailSum(3, 0))
```

Output?
A) 3
B) 6
C) 0
D) 9

Learning: Tail recursion builds result via accumulator, useful for space optimization.

MCQ 2: Tail Factorial

```
function tailFact(n, acc):
    if n == 0:
        return acc
    return tailFact(n - 1, acc * n)

print(tailFact(4, 1))
```

Output?
A) 24
B) 12
C) 6
D) 10

MCQ 3: Count Down Tail

```
function count(n):
    if n == 0:
        return
    print(n)
    count(n - 1)

count(3)
```

Output?
A) 3 2 1
B) 1 2 3
C) 0 1 2
D) 2 1 0

Learning: Although `print(n)` is before the call, the recursive call is last → still tail-recursive.

MCQ 4: Tail Recursion for Power

```
function power(base, exp, acc):
    if exp == 0:
        return acc
    return power(base, exp - 1, acc * base)

print(power(2, 3, 1))
```

Output?
A) `6`
B) `8`
C) `9`
D) `16`

MCQ 5: Tail Recursion with Condition

```
function skipEven(n):
    if n > 5:
        return
    if n % 2 != 0:
        print(n)
    skipEven(n + 1)

skipEven(1)
```

Output?
A) `1 2 3 4 5`
B) `1 3 5`
C) `2 4`
D) `1 2 3 5`