P4

- *Name & purpose*:
  → counts how many times the nested loops execute by combining linear and logarithmic iteration patterns.

- *Inputs*:
  → **n**: integer (controls the range of both the outer and inner loops)

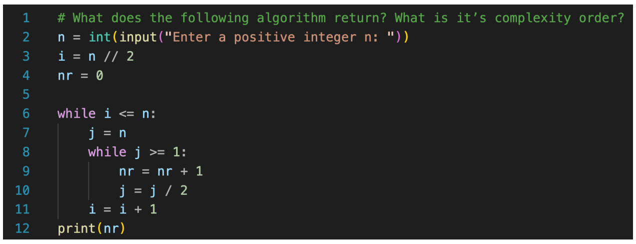
- *Outputs*:
  → **nr**: integer (total number of iterations performed)

- *Preconditions*:
  → **n** is a positive integer

- *Postconditions*:
  → returns the total count of all inner-loop executions

- *High-level idea*:
  → the outer loop runs from *n/2* up to *n*
  → for each outer step, the inner loop repeatedly divides *j* by 2 until it drops below 1
  → the total count increases each time the inner loop runs.

- *Pseudocode*:



```
# What does the following algorithm return? What is it’s complexity order?
n = int(input("Enter a positive integer n: "))
i = n // 2
nr = 0

while i <= n:
    j = n
    while j >= 1:
        nr = nr + 1
        j = j / 2
    i = i + 1
print(nr)
```

- *Complexity*:
  → time: Θ(n log n) → outer loop Θ(n), inner loop Θ(log n)
  → space: Θ(1)

- *Correctness sketch*:
  → each outer loop runs for all *i* between *n/2* and *n*
  → each inner loop halves j until 1, so total count correctly tracks all iterations

- *Edge cases*:
  → n = 1 → outer loop runs once; inner loop runs once
  → n = 0 or negative → invalid input (loop not executed)
  → very large numbers → may overflow *nr*