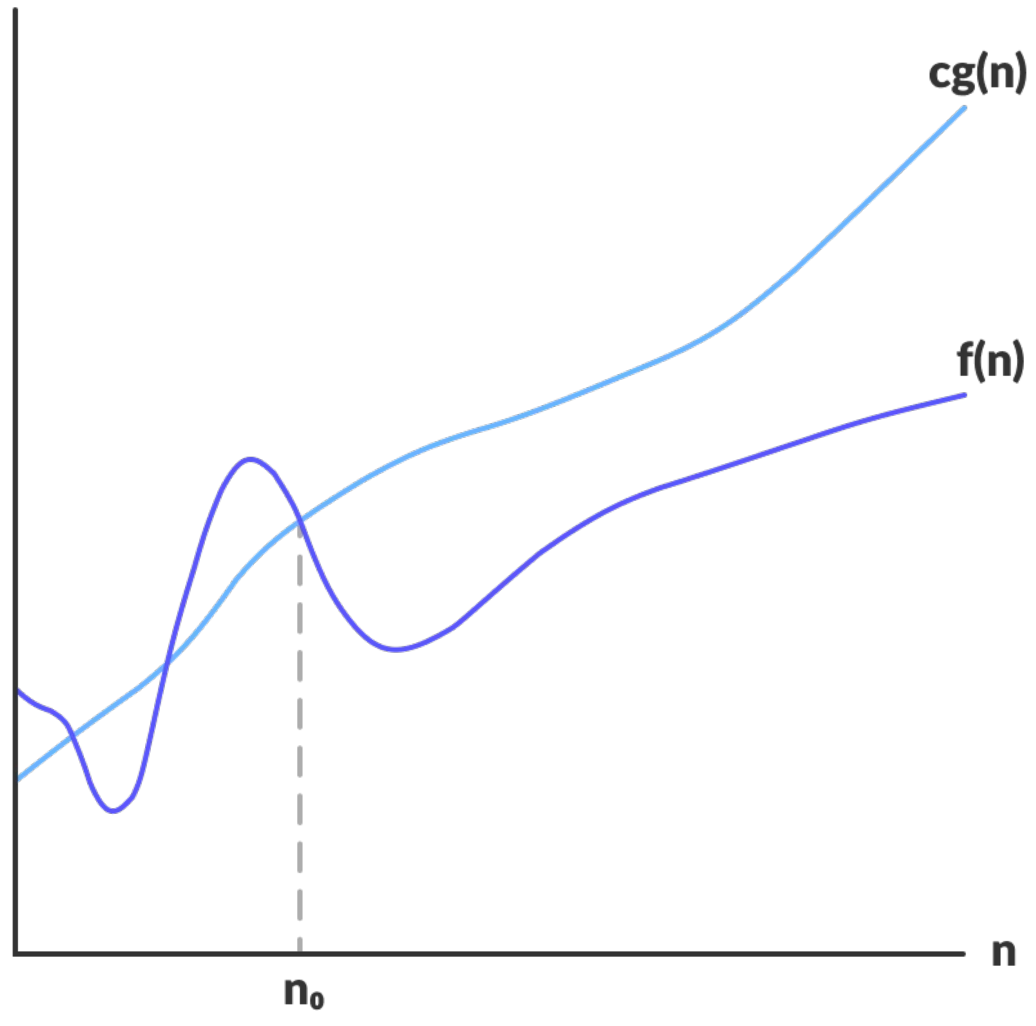
Complexity of Algorithms

Big-O, little-o

$f \in O(g)$

$f \in o(g)$

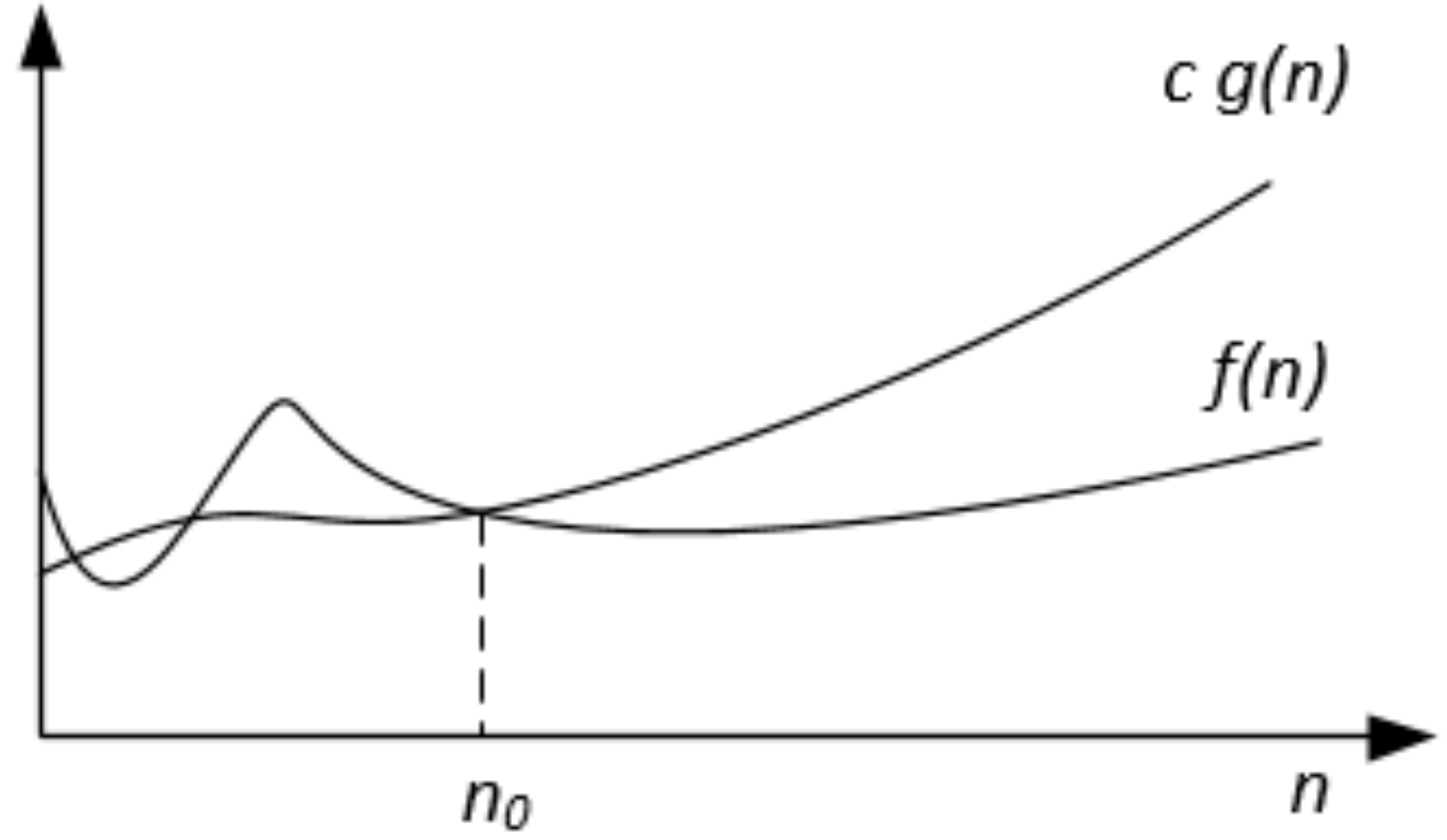


https://www.devopsschool.com/blog/complete-tutorial-on-big-o-big-oh-notation/

Formal definition

$$f \in O(g) \Leftrightarrow \exists c > 0, \exists n_0 > 0 : \forall n \geq n_0,\ 0 \leq f(n) \leq c \cdot g(n)$$

$$f \in o(g) \Leftrightarrow \forall c > 0, \exists n_0 > 0 : \forall n \geq n_0,\ 0 \leq f(n) \leq c \cdot g(n)$$



Big-O example

https://devtut.github.io/algorithm/algorithm-complexity.html#comparison-of-the-asymptotic-notations

Analogy to equality

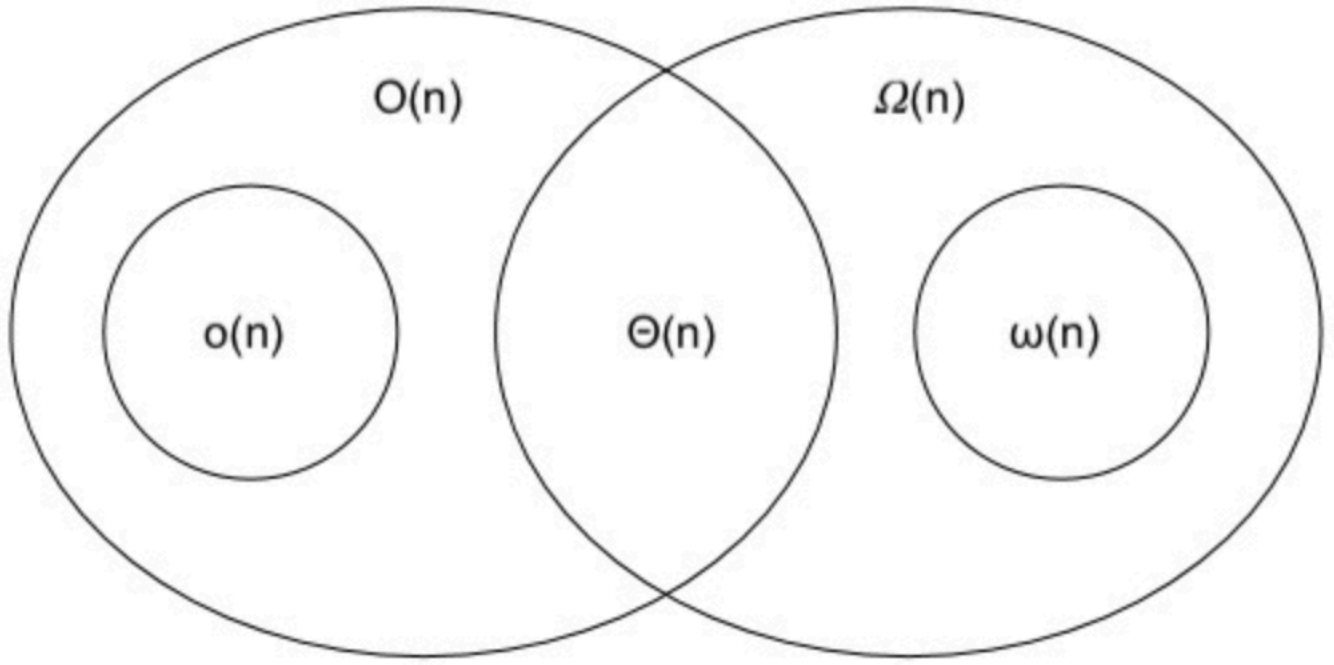
O(n): grows no faster than n

o(n): grows strictly slower than n

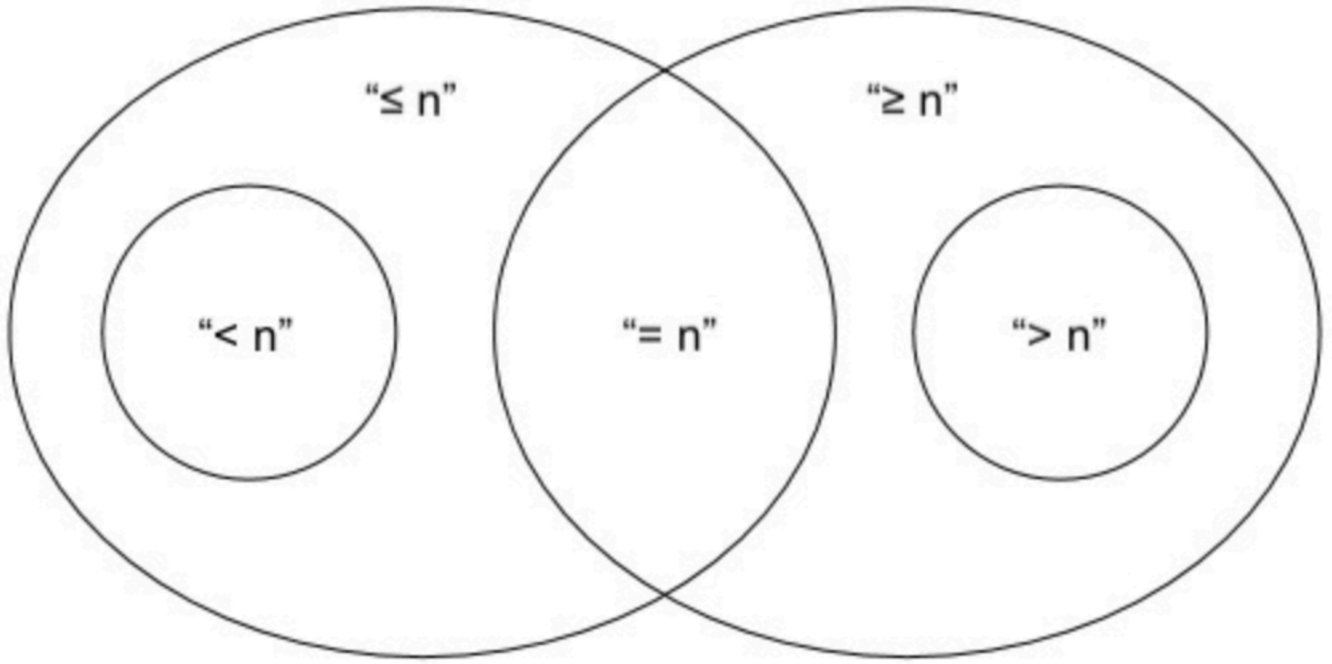
Ω(n): grows faster than n

ω(n): grows strictly faster than n

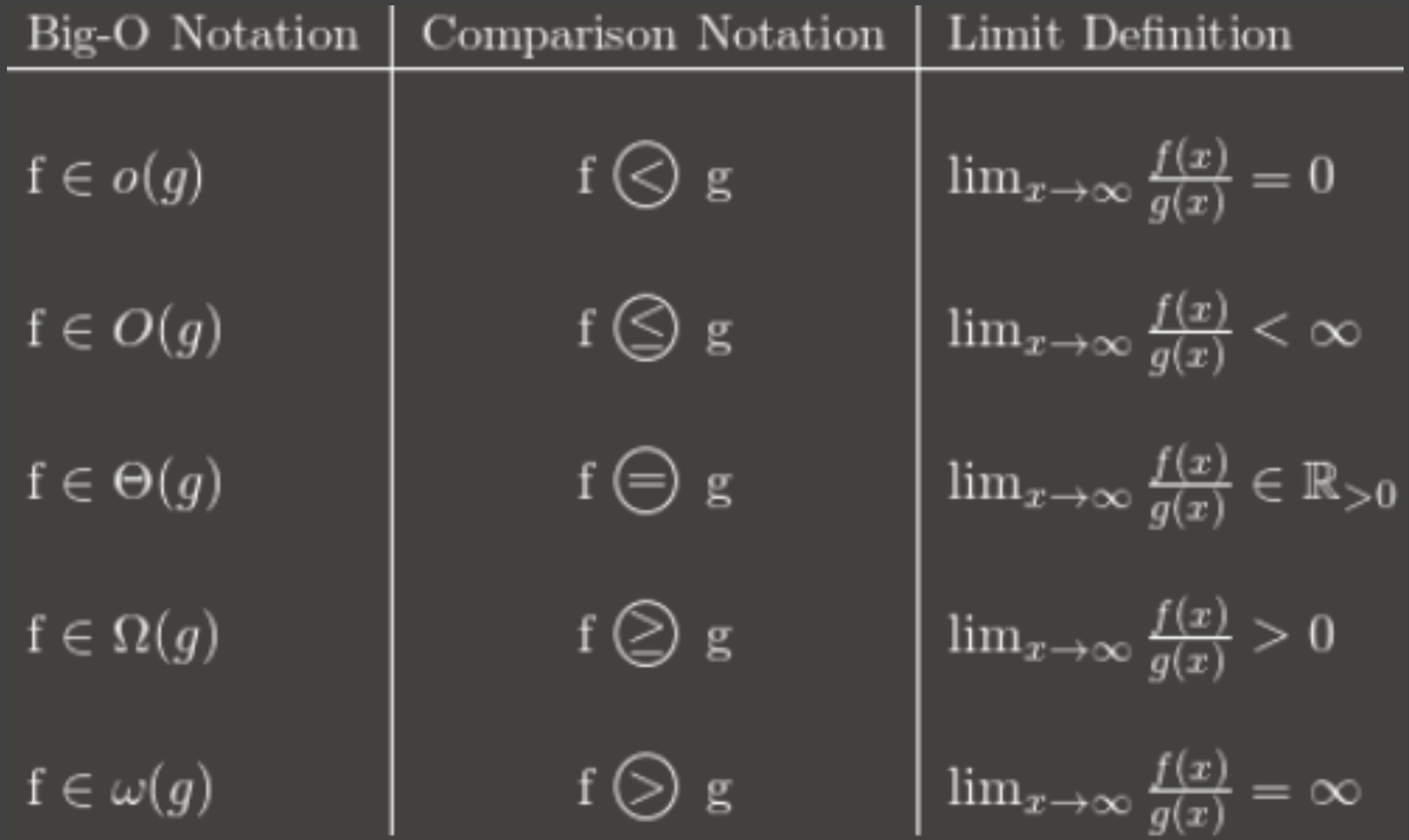
Θ(n): grows as fast as n



https://devtut.github.io/algorithm/algorithm-complexity.html#comparison-of-the-asymptotic-notations



In limit notation



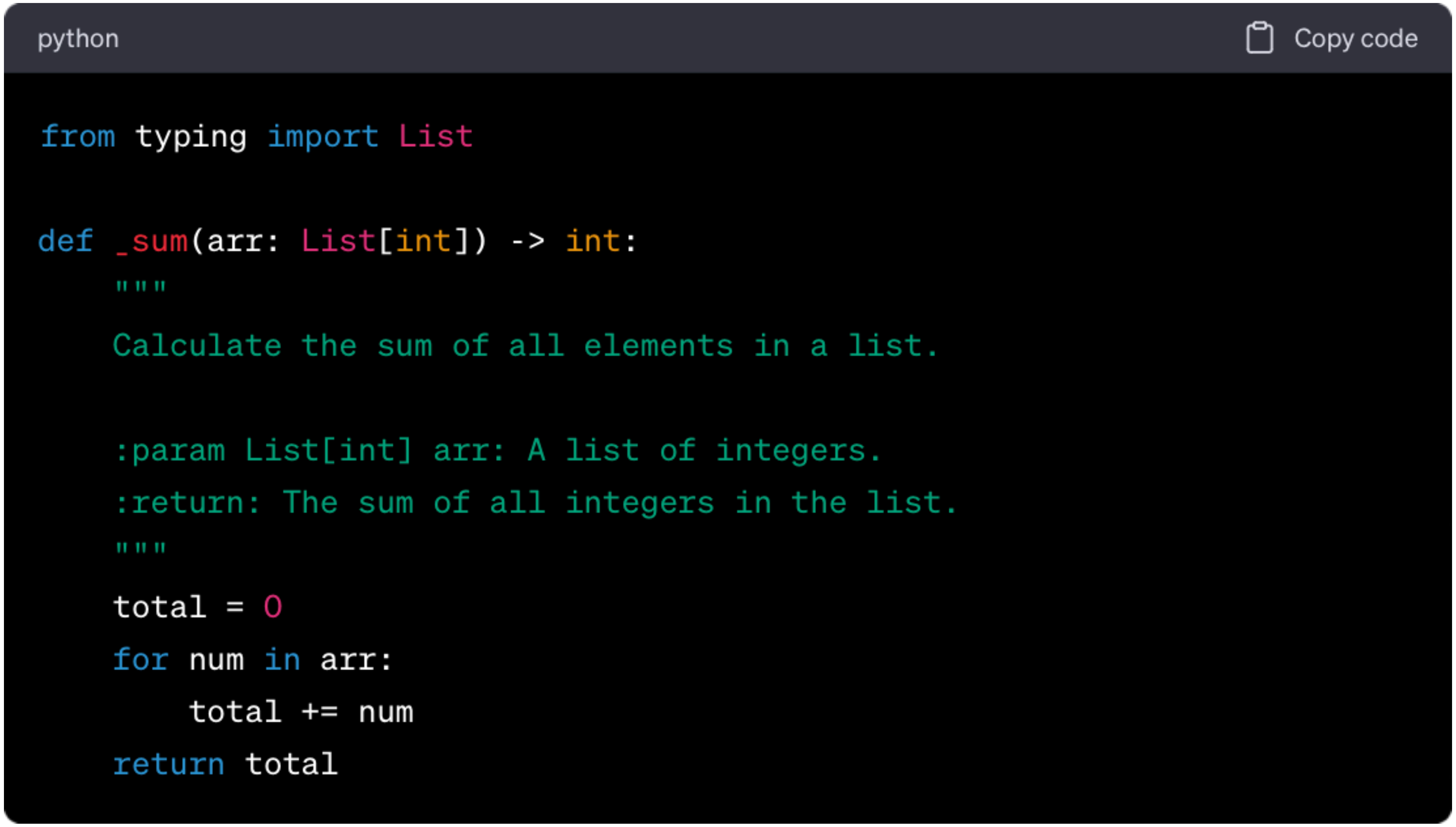
Big-O Notation	Comparison Notation	Limit Definition
$f \in o(g)$	$f \textcircled{$<$} g$	$\lim_{x\to\infty} \frac{f(x)}{g(x)} = 0$
$f \in O(g)$	$f \textcircled{$\leq$} g$	$\lim_{x\to\infty} \frac{f(x)}{g(x)} < \infty$
$f \in \Theta(g)$	$f \textcircled{$=$} g$	$\lim_{x\to\infty} \frac{f(x)}{g(x)} \in \mathbb{R}_{>0}$
$f \in \Omega(g)$	$f \textcircled{$\geq$} g$	$\lim_{x\to\infty} \frac{f(x)}{g(x)} > 0$
$f \in \omega(g)$	$f \textcircled{$>$} g$	$\lim_{x\to\infty} \frac{f(x)}{g(x)} = \infty$

https://stackoverflow.com/questions/25882333/if-something-is-little-o-of-fn-is-it-also-big-o-of-fn

Profiling code

Quiz

Code example 1

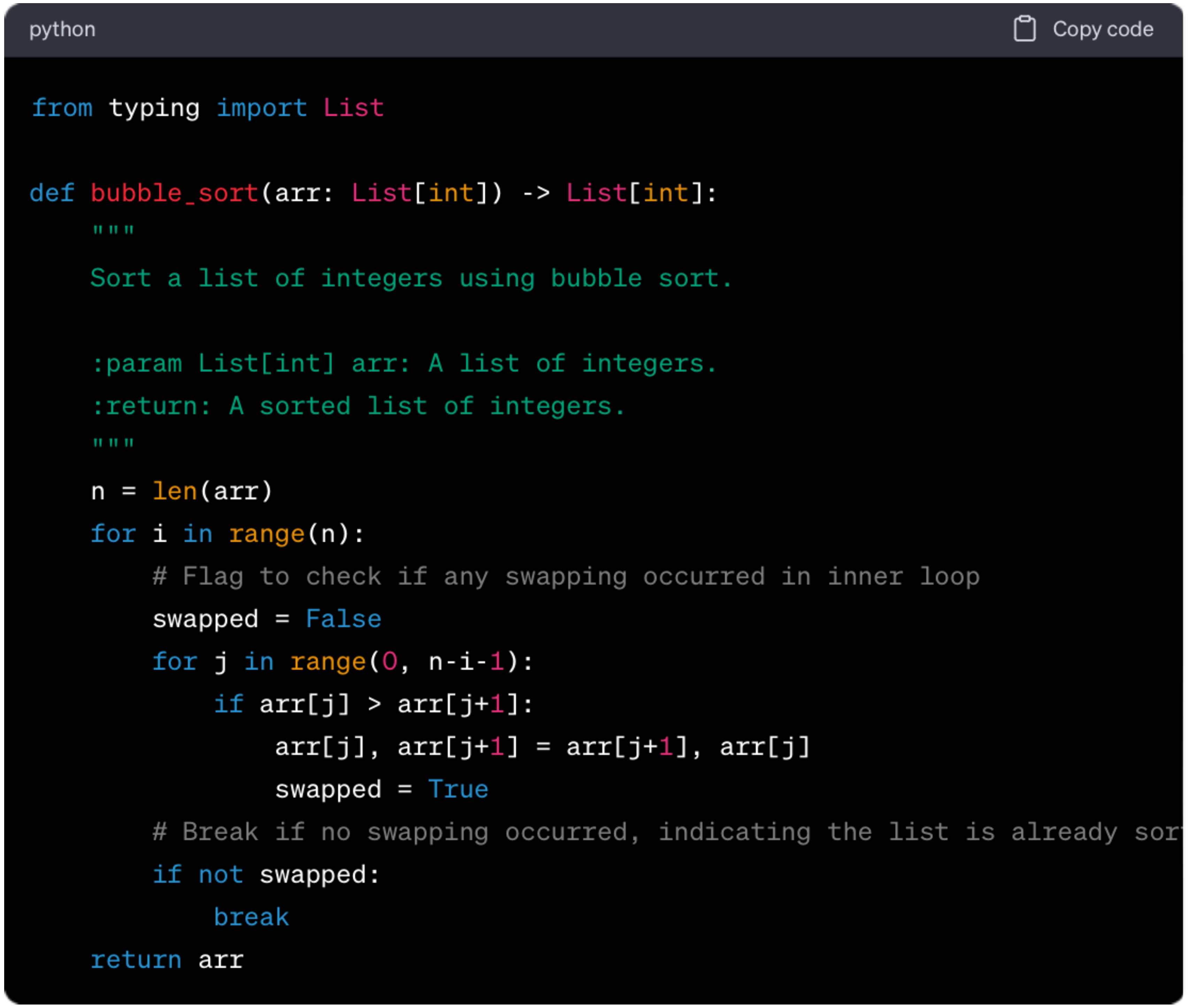

python
Copy code

```python
from typing import List

def _sum(arr: List[int]) -> int:
    """
    Calculate the sum of all elements in a list.

    :param List[int] arr: A list of integers.
    :return: The sum of all integers in the list.
    """
    total = 0
    for num in arr:
        total += num
    return total
```

Code example 2

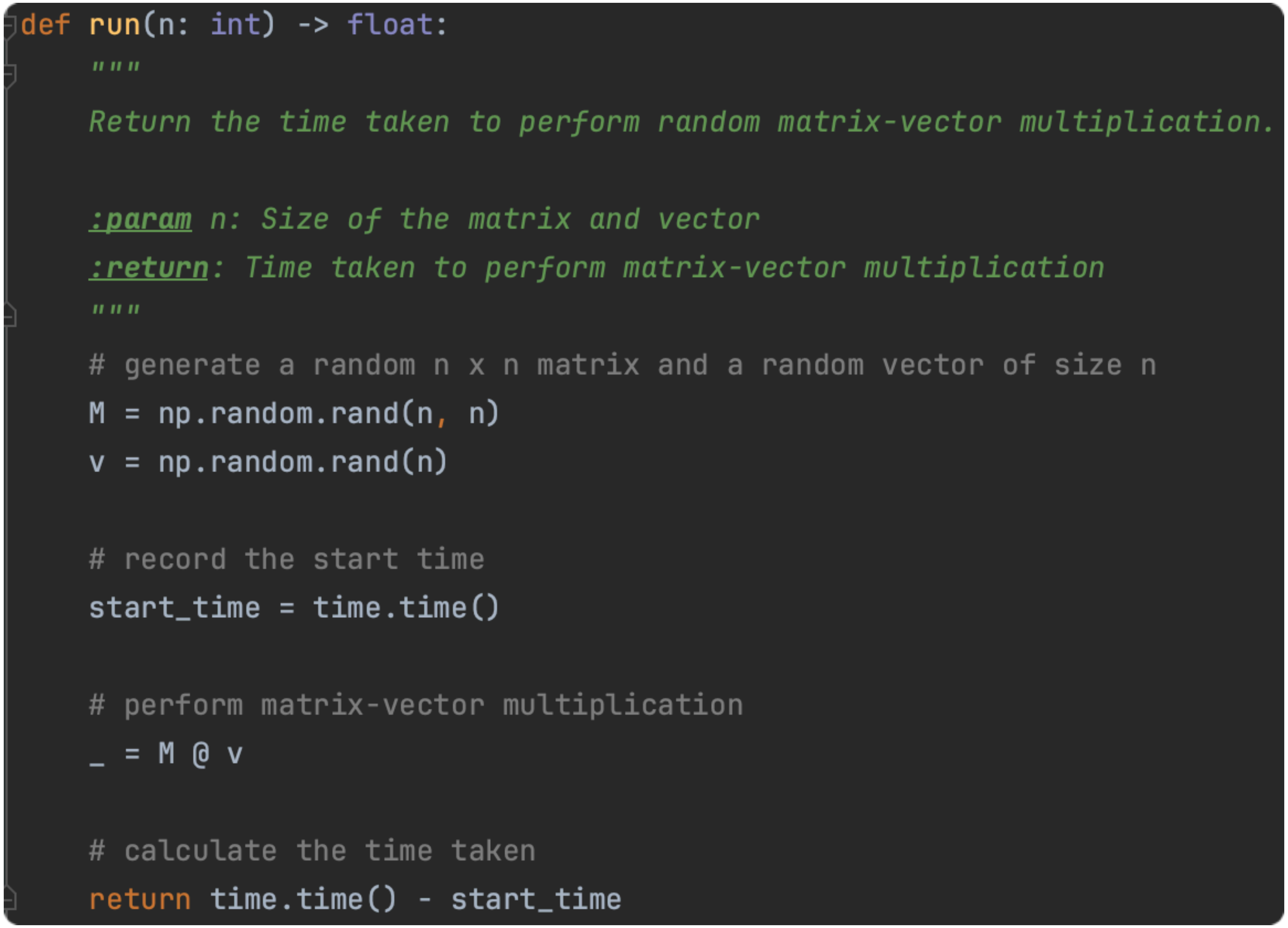


```python
from typing import List

def bubble_sort(arr: List[int]) -> List[int]:
    """
    Sort a list of integers using bubble sort.

    :param List[int] arr: A list of integers.
    :return: A sorted list of integers.
    """
    n = len(arr)
    for i in range(n):
        # Flag to check if any swapping occurred in inner loop
        swapped = False
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]
                swapped = True
        # Break if no swapping occurred, indicating the list is already sor
        if not swapped:
            break
    return arr
```

Question



```python
def run(n: int) -> float:
    """
    Return the time taken to perform random matrix-vector multiplication.

    :param n: Size of the matrix and vector
    :return: Time taken to perform matrix-vector multiplication
    """
    # generate a random n x n matrix and a random vector of size n
    M = np.random.rand(n, n)
    v = np.random.rand(n)

    # record the start time
    start_time = time.time()

    # perform matrix-vector multiplication
    _ = M @ v

    # calculate the time taken
    return time.time() - start_time
```