Example 1: Measuring Time Complexity

In this lesson, we are going to learn how to compute the running time complexity of an algorithm that involves loops.

We'll cover the following

In the previous lesson, we calculated the running time complexity of a very basic Python program. Lets now calculate the running time complexity of a more complex program. We will split the code into individual operations and then compute how many times each is executed.

Simple For Loop of Size `n`

Here is an example of a simple loop of size ***n***:

```python
n = 10  # just as an example, n can be anything
sum = 0
for var in range(n):
    sum += 1

print(sum)

```

Operation	Number of executions
`n = 10`	1
`sum = 0`	1
`range(n)`	1
`var=0`	1
`var=1`	1
`var=2`	1
...	
`var=n-1`	1
`sum+=1`	$3 \times n$
`print(sum)`	2

> Note that while `range(n)` executes only once, its execution cost is n.
>
> This is because it creates a list of values from 0 to n - 1.

Expand to recall range(n)

Running Time Complexity#

After counting how many times each operation is executing, we will just add all of these counts to get the time complexity of this program.

$$\text{Time complexity} =$$

$$1 + 1 + \text{n} + (1 + 1 + 1 + ... + 1) + 3\text{n} + 2$$

$$\Rightarrow\ 2 + \text{n} + \text{n} + 3\text{n} + 2$$

$$\Rightarrow 5n + 4$$

In the next lesson, we will look at another example of a program containing nested loops and compute its running time complexity.

Back
Comparing Algorithms

Next
Example 2: Measuring Time Complexity

Completed

Report an Issue