Solution Review: Big O of Nested Loop with Subtraction

This review provides a detailed analysis of the time complexity of the Nested Loop with Subtraction problem!

We'll cover the following

Solution

```
n = 10  # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n, 1, -3):
    print(pie)
    for j in range(n, 0, -1):
        sum += 1

print(sum)

```

The variable `var` gets set to n then $n-3, n-(2\times 3), n-(3\times 3), \cdots, 3$ in the outer loop. So the loop runs $\frac{n}{3}$ times. Try the following,

```
n = 12
print(len(range(n, 0, -3)))  # The length is n/3

```

Have a look at the following slides for a more detailed derivation of the time complexity

```
n = 10 # n can be anything, this
is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

0

Let's dry run this code to calculate its running time complexity.


```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

1

initializing `n`. This costs us one unit of time.

```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

2

Initializing `sum`. This also costs us one unit of time.


```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

3

Initializing `pie`. This, again, costs us one unit of time.

```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

3+n/3

As explained in the first challenge and above, range will take n/3 units of time here.


```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

3+n/3+n/3

`var` will be assigned to be equal to an element in the list generated by the range function at each iteration so, there will be n/3 assignments

```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

3+n/3+ n/3+n/3

printing takes one unit of time and pie will print once for every iteration of the outer loop. For n/3 iterations the total time will be n/3.

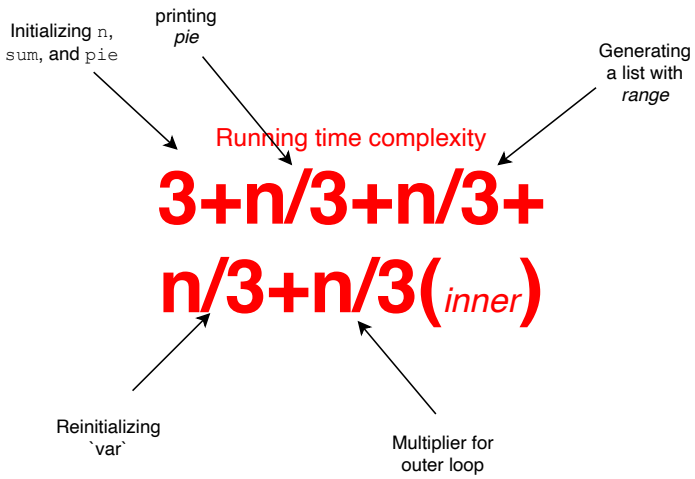

```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

3+n/3+n/3+ n/3+n/3(*inner*)

Now, let's calculate the time complexity of the inner loop. The inner loop will run n/3 times so we've put a placeholder for it for now.



Here's a recap of what we have so far.


```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

inner

Now lets calculate the time complexity of the inner loop.

```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

n

Creating a new list with range takes n units of time because the list will have n elements


```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

n+n

j will get assigned to a new value n times

```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

n+n+2n

Addition of 1 to the current value of `sum` and then assigning that value to the variable `sum` takes two units of time.


```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

n+n+2n

Total time complexity of the inner loop. We'll plug this running time complexity into the total time complexity computed so far.

```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

3+n/3+n/3

+n/3+n/3(n+n+2n)

Total time complexity

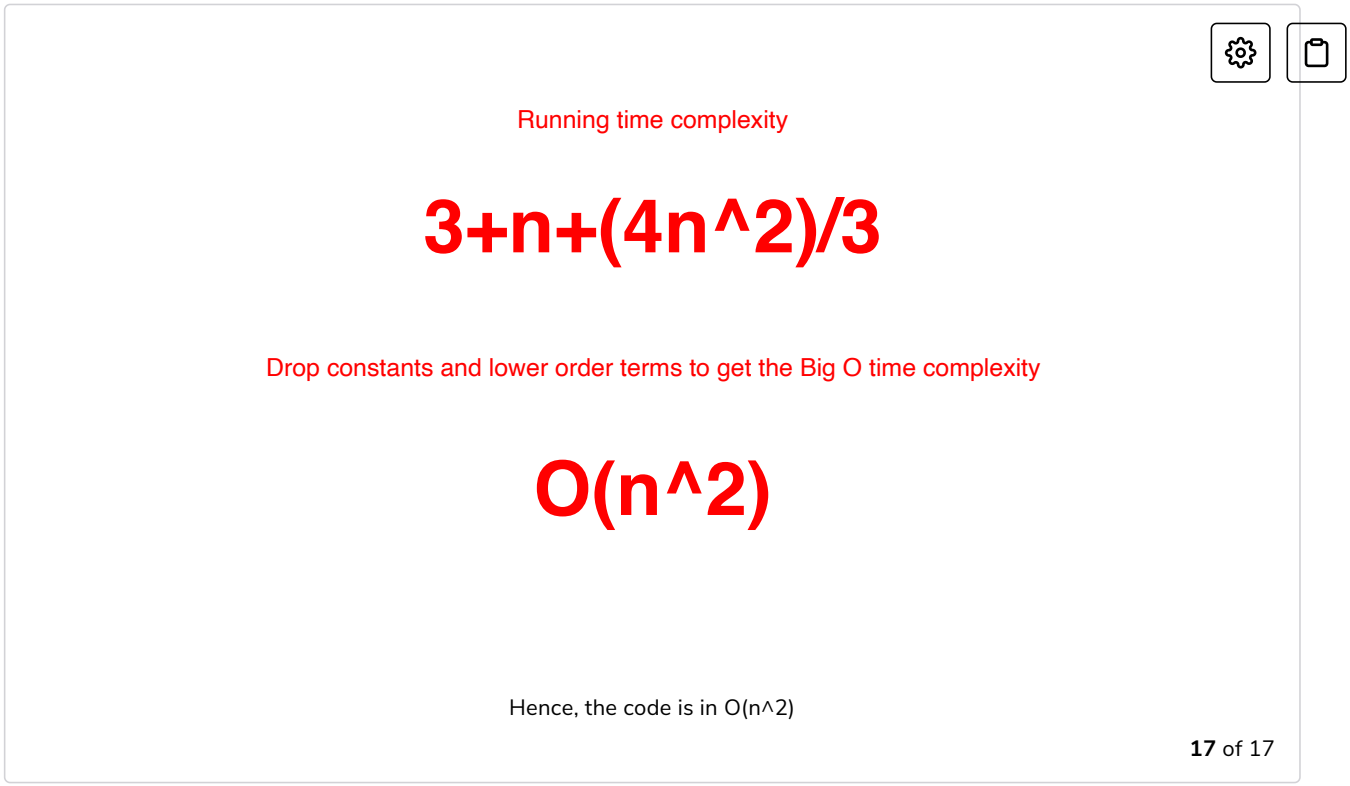

```
n = 10 # n can be anything, this is just an example
sum = 0
pie = 3.14
for var in range(n,1,-3):
  print(pie)
  for j in range(n,0,-1):
    sum+=1

print(sum)
```

Running time complexity

3+n+(4n^2)/3

Simplifying



Big O time complexity: $\mathbf{O(n^2)}$

Back
Challenge 2: Big O of Nested Loop wit...
Next
Challenge 3: Big O of Nested Loop wit...
Completed
Report an Issue
Ask a Question (https://discuss.educative.io/tag/solution-review-big-o-of-nested-loop-with-subtraction__introduction-to-complexity-measures__data-structures-for-coding-interviews-in-python)