COMPUTER SCIENCE

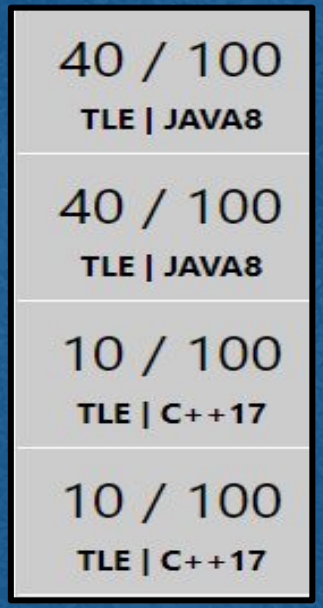
CLUB

Intro to Competitive Programming (Pt. 2)

Edward

Status Codes

https://dmoj.ca/about/codes/



Subtask 2 [0/5 points] [0/9 tests passed]		
Test 15	Time limit exceeded	2.1 seconds [Terminated]
Test 16	Skipped Test	0.1 seconds
Test 17	Skipped Test	0.1 seconds

Example - how many operations?

```python
n = int(input())
ans = 0
for i in range(n):
    for j in range(n):
        ans += 1
print(ans)
```

Better optimized code

```
1    n = int(input())
2    print(n**2)
```

Big Θ Notation

How fast/efficient your algorithm is in accordance with the input

Analyzing Big Θ

```
1  n = int(input())
2  ans = 0
3  for i in range(n):
4      for j in range(n):
5          ans += 1
6  print(ans)
```

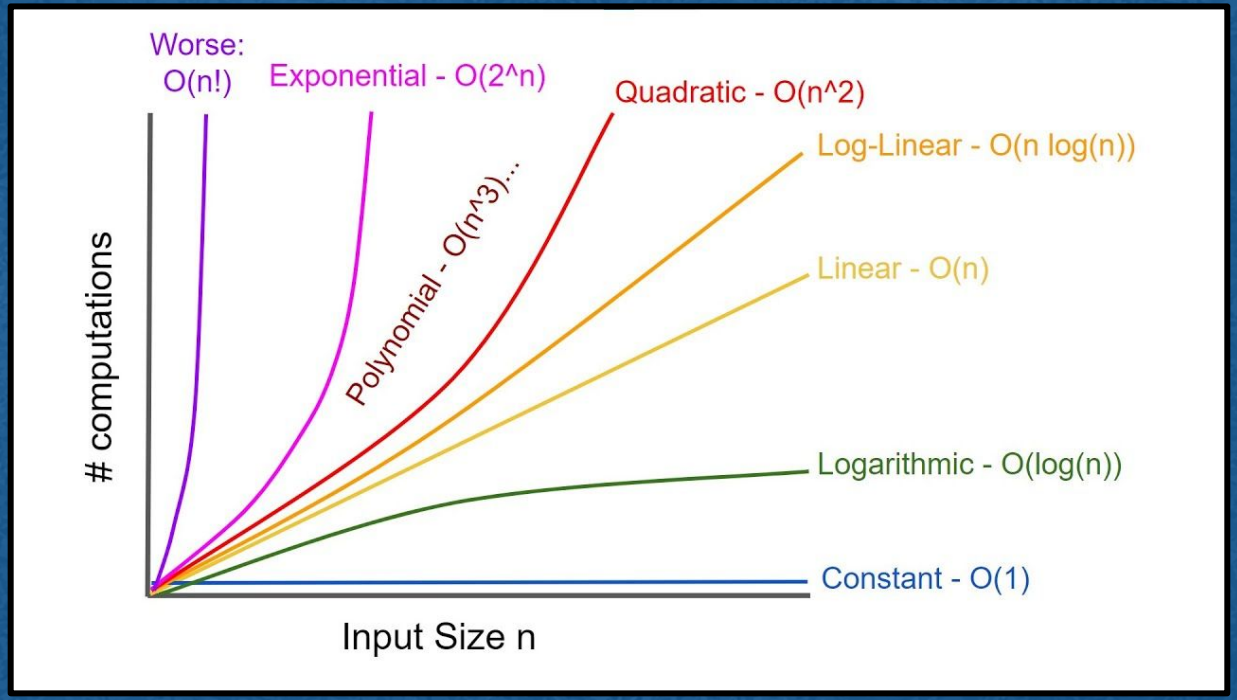
$O(n^2)$

Analyzing Big Θ

```
1    n = int(input())
2    print(n**2)
```

O(1)

Big Θ Visualization



Big Θ "Rules"

$$O(2n) = O(n)$$

$$O(99n + 200) = O(n)$$

$$O(n! + n^{88}) = O(n!)$$

$$O(mn) = O(mn)$$

$$O(m+n) = O(\max(m, n))$$

Analyzing Big Θ

```python
n = int(input())
ans = 0
for i in range(n):
    for j in range(n):
        ans += 1

for i in range(n):
    print(ans)
```

$$O(n^2+n) = O(n^2)$$

Analyzing Big Θ

```python
dict = {
    '1': True,
    '2': True,
    '4': True
}
set = {'1', '2', '4'}
list = ['1', '2', '4']

print('2' in dict)
print('2' in set)
print('2' in list)
```

Dictionary: O(1)

Set: O(1)

List: O(n)

Constant Time: Θ(1)

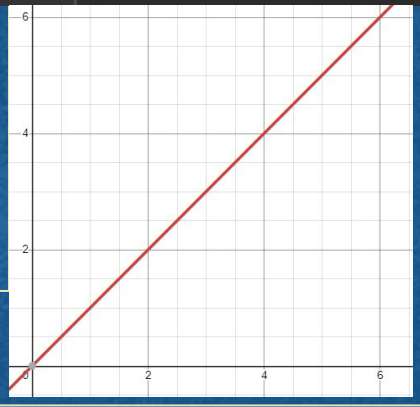
- Does not scale with input
- Examples:
  - Math
  - Taking input/printing
  - Declaring variables
  - If statements
  - Reading numbers or list indices

```
1    n = 12234
2    a = 4312
3    b = [12341, 444, 555]
4    if n > 300:
5        print(n + a / b[0])
6    else:
7        print(b[2])
```

Linear Time: Θ(n)

- Increases linearly with input
- Examples:
  - For loops
  - Python .count()
  - Python x in list
  - Looping an array with length n
  - Searching lists for an element
  -

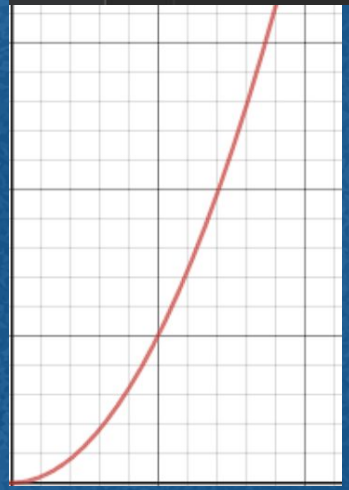
```python
list = [1, 2, 3, 45]
n = 1
for i in range(n):
    pass
print(list.count(2))
print(1 in list)
```



Quadratic Time: $\Theta(n^2)$

- Increases exponentially with input
- Examples:
  - a for loop inside of a for loop
  - O(n) inside a for loop

```
1   n = int(input())
2
3   for i in range(n):
4       for j in range(n):
5           print("hi!!!")
```



Other complexities

- Other polynomial times ($O(n^3)$, $O(n^{15})$, $O(n^{932})$)
  - More nested for loops
- $O(\log n)$: splitting in half each time
  - Binary search
  - Most sort algorithms are $O(n \log n)$
- $O(2^n)$: doubling each time
  - Brute force/trees
- $O(\sqrt{n})$: certain math situations
  - Checking if prime
- $O(n!)$: brute force combinations

Time Complexity to Input Chart

Constraints

$1 \leq N \leq 10^{12}$

CCC will run ~10^7 operations per second

Range	Time Complexity
$n \leq 10$	$\mathcal{O}(n!)$
$n \leq 20$	$\mathcal{O}(2^n)$
$n \leq 50$	$\mathcal{O}(n^5)$
$n \leq 100$	$\mathcal{O}(n^4)$
$n \leq 500$	$\mathcal{O}(n^3)$
$n \leq 3000$	$\mathcal{O}(n^2 \cdot \log n)$
$n \leq 5000$	$\mathcal{O}(n^2)$
$n \leq 10^5$	$\mathcal{O}(n \cdot \log n) \sim \mathcal{O}(n \cdot \sqrt{n})$
$n \leq 10^6$	$\mathcal{O}(n \cdot \log n) \sim \mathcal{O}(n)$
$n \leq 10^{12}$	$\mathcal{O}(\sqrt{n})$
$n \leq 10^{18}$	$\mathcal{O}(\log n)$

Calculating if your code will pass

1. Calculate time complexity (e.g. O(n), $O(n^2)$)
2. Substitute the maximum of the input into your time complexity
3. Is the result over 10 million?

e.g. O(n) = O(1000000) < 10 million ⟶ Pass!

O(n^2) = O(1000000^2) = O(1 trillion) > 10 million ⟶ TLE!

Practice Problems:

Beginner: https://dmoj.ca/problem/dmpg17s1
(try to find an O(nlogn) solution
Intermediate: https://dmoj.ca/problem/ccc22s2
(analyze big O - use dictionaries/hashmaps)
Advanced: https://dmoj.ca/problem/dmopc20c2p2
or https://dmoj.ca/problem/ccc21s2