Time Complexity 2

AGENDA:

- Asymptotic Analysis
- Big O
- Issues with Big O
- Space complexity
- TLE

Comparing Algorithms

Given an array, sort it.

Sachin

Supersort

10 sec

Macbook M2

Pranay

Awesome-sort

15 sec

Old Laptop – Windows 98

Both on Mac M2

10 sec

Python

7 sec

C++

Both on C++

5 sec

Hill station

7 sec

Close to volcano

Both on hill station

5 sec

5 sec

Same

Execution Time – Not a reliable measure of performance for algo's

→ It depends on external factors

→ SW / HW / Environment

```
for (i=0; i<N; i++) {          → N iterations
    ==
}
```

No of iterations is independant of external factors

Angeline	Shubhankur
$100 \log_2 N$	$N/10$

Before 3550 Shubhankur prefered

After 3550 Angeline prefered

Overall – $\boxed{\text{Angeline's algo is more efficient}}$

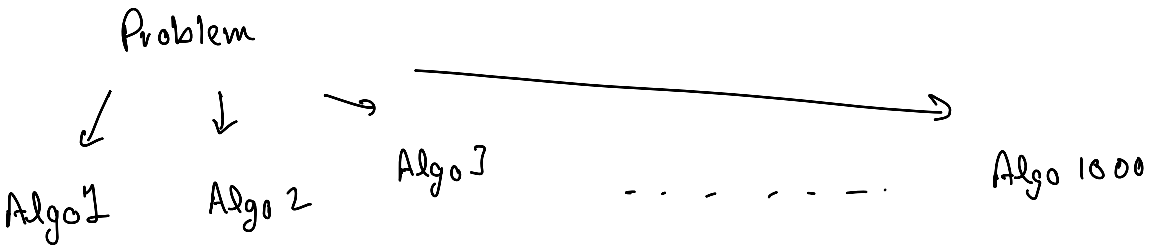
Hotstar – 2×10^8 people

Google – 10^8 search results in 0.012 s

Despacito – 7 B +

Huge data

Increasing



Problem

Algo 1

Algo 2

Algo 3

. . . .

Algo 1000

Graph method is not possible to compare.

Asymptotic Analysis of Algorithms

Observing performance of algorithms for very large inputs.

↳ Big O

Comparisons using Big O notation

Angeline

$100 \log_2 N$

↓

$\underbrace{O(\log_2 N)}_{\text{Better}}$

Shubhankur

$N/10$

↓

$O(N)$

1) Calculate the no of iterations

2) Neglect all lower order terms

3) Neglect constant coefficients

Why neglect lower order terms ?

Lower order

Algo : $N^2 \boxed{+ 10N}$

% contribution of lower order term

$N=100 \rightarrow 10^4 + 10^3$ $\qquad \frac{10^3}{10^4+10^3} \times 100 = 9.09\%$

$N=10^4 \rightarrow 10^8 + 10^5$ $\qquad \frac{10^5}{10^8+10^5} \times 100 = 0.1\%$

Contribution of lower order terms is significantly small for larger inputs

Negligible

Why neglect constant coefficient ?

Manaswini	Sufyan
$10 \log_2 N$	N
$10^2 \log_2 N$	N
$10^3 \log_2 N$	N
$10^4 \log N$	$10N - 10^3$

Issues with Big-O

Issue 1

	Ajith	AbinDas	More efficient
	100N	N^2	
N= 50	100×50	50 ×50	AbinDas
N= 80	100×80	80 ×80	AbinDas
N=100	100×100	100×100	Same
N= 120	100 ×120	120×120	Ajith
N=150	100×150	150×150	Ajith

↓ $O(N)$

↓ $O(N^2)$

Big O doesn't give you complete picture.

BigO comparison is not true for all input sizes.

Issue 2

Vignesh	Manikanta
$N^2 + 10N$	$2N^2 + 5N$
↓	↓
$O(N^2)$	$O(N^2)$

Issue : Comparison says → Both are same

$N^2 + 10N$ $2N^2 + 5N$

Note : Big O will solve 99% of your problems

Linear Search

```
for (i=0; i<N; i++)
        if ( arr[i] = = x )
                return true
return false
```

Best - O(1)

Worst - O(N)

Default - Worst

Space Complexity

Break till 10:14 AM

Amount of extra space taken by your algorithm.

Extra space

Q1

```
func (int N) {
        int x = N
        int y = x + x
        int z = x + y
}
```

int - 4 bytes

3 x 4 = 12 bytes

Constant ↓ O(1)

Q2

Quiz 1

```
func (int N) {
    int x = N
    int y = x^2
    int z = x+y

    arr[] = int [N]   ← Array of integers
                 ↑
             Size of array
}
```

$4B + 4B + 4B + 4N$ bytes

$= (12 + 4N)$ bytes

↓

$O(N)$

Q3

Quiz 2

```
func (int N) {
    int x = N          - 4 bytes
    int y = x^2        - 4 bytes
    int z = x+y        - 4 bytes
                                       4N bytes
    arr [] = int [N]          ← 1D Array of size N
    l [][] = int [N][N]       ← 2D Array of NxN     ← 4N^2 bytes
}
```

Total = $(4+4+4+4N+4N^2)$ bytes

↓

$O(N^2)$

Q4

Array of int ↓   Size of array ↓

```
sumOfArray ( int  arr[],  int  N) {

      sum = 0                              - 4B

      for ( i=0 ; i<N ; i++) {             - 4B

            sum  =  sum  +  arr[i]

      }

      return  sum
}
```

Total extra space

= (4 +4) bytes

= 8 bytes

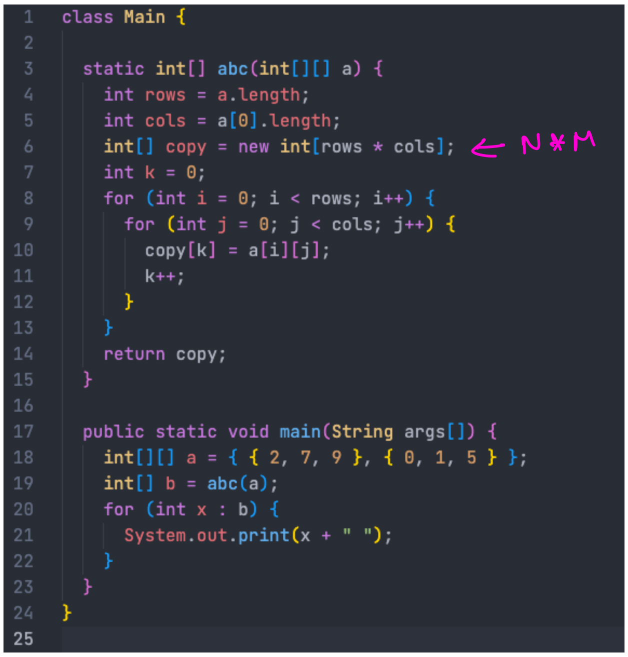
↓

O(1)

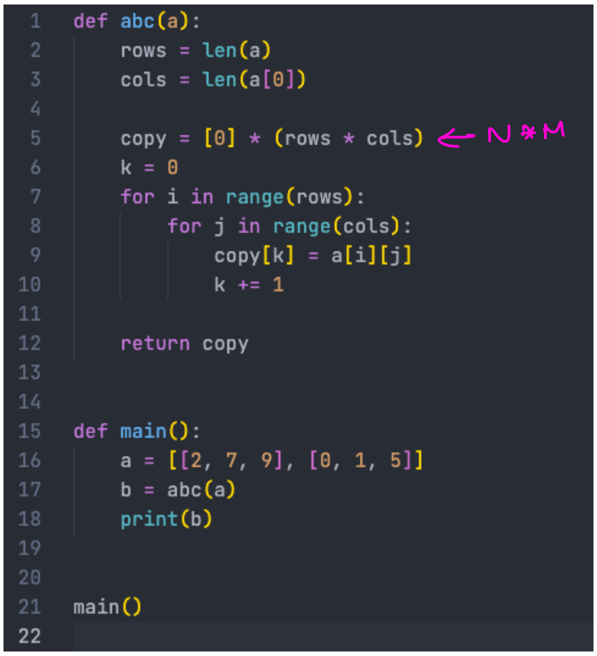
Q5

Quiz 3

What is the space complexity of the abc() function of the following code snippet, assuming the input matrix is of size N*M?



```java
class Main {

  static int[] abc(int[][] a) {
    int rows = a.length;
    int cols = a[0].length;
    int[] copy = new int[rows * cols];  ← N*M
    int k = 0;
    for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
        copy[k] = a[i][j];
        k++;
      }
    }
    return copy;
  }

  public static void main(String args[]) {
    int[][] a = { { 2, 7, 9 }, { 0, 1, 5 } };
    int[] b = abc(a);
    for (int x : b) {
      System.out.print(x + " ");
    }
  }
}
```



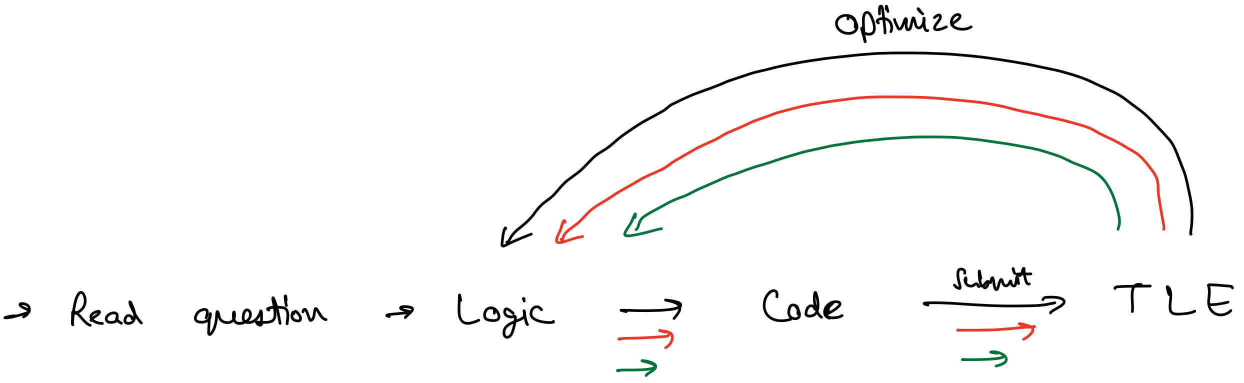
```python
def abc(a):
    rows = len(a)
    cols = len(a[0])

    copy = [0] * (rows * cols)  ← N*M
    k = 0
    for i in range(rows):
        for j in range(cols):
            copy[k] = a[i][j]
            k += 1

    return copy


def main():
    a = [[2, 7, 9], [0, 1, 5]]
    b = abc(a)
    print(b)


main()
```

Extra space O(N*M)

TLE - Time Limit Exceeded

Google Contest → 2hrs

3 Questions



// Check whether TLE without even writing a single line of code

Online Judges → Their servers → 1 GHz

↓

10^9 operations / second

Time Limit - 1 sec

At max, our code can have 10^9 operations

Operation – Variable declaration, +, -, ÷, x, %

```
          +1       +1        +1
for (i=0;    i< N;    i=i+1) {
            +1     +1
    if ( i%2   ==1)
        print(i)  +1
}
```

Loop

Operations

1 + 5N

Assumption

1 iteration → 10 operations

10^8 iterations → 10^9 operations

At max, our code can have

10^8 iterations

(Approx)

Process to solve

1) Read & understand the question

2) Logic ←

3) Checking correctness　　If TLE

Try it for multiple testcases

4) Check if TLE occurs

↓ If no TLE

5) Write code & execute

Importance of constraints

Eg

$1 <= N <= 10^6$

$1 <= arr[i] <= 10^8$

Algo - $O(N^2)$ time

$N = 10^6$ → 10^{12} iterations

TLE

1 sec → 10^8 iterations

Eg

$1 <= N <= 5 \times 10^3$

Algo - $O(N^2)$ ← Not optimized

$N = 5 \times 10^3$

$N^2 \rightarrow (5 \times 10^3)^2 = 25 \times 10^6 < 10^8$

Will work

No TLE

Eg $1 <= N <= 5 \times 10^2$

Rare

Algo : $O(N^3)$ time

$N = 5 \times 10^2$

$N^3 = 125 \times 10^6$

$= 1.25 \times 10^8$ iterations

$\cong 10^8$ iterations

Can't be said

It might work

It might not work

Corner Case

Doubts

Thank you

C++ → Compiled language

Python → Interpreted language

Python → C++ →

$\underline{10^8}$ N

↑

Cannot be ignored

$1 <= N <= 10^8$

```
for i → [1,n]
    for j → [1, $3^i$]
```

i	j	Iterations
1	$1 \to 3^1$	3 +
2	$1 \to 3^2$	3^2 +
3	$1 \to 3^3$	3^3 +
⋮		⋮ +
N	$1 \to 3^N$	3^N

Total =

$3 + 3^2 + 3^3 + \ldots 3^N$

GP formula

10 operations → 1 iteration

1 operation → $\frac{1}{10}$ iteration

10^9 operations → $\frac{1}{10} \times 10^9$ iterations

$= 10^8$ iterations

$10000 \log x$ $10x - 10000$

for (i=1 ; i<=n ; i+=2)

<=N

i → 1, 3, 5, 7, 9, 11 . . . . .

Approx - $\frac{N}{2}$ iterations → $O(N)$

$$\frac{N}{2} = \frac{1}{2} \times N$$

↑
constant coefficient

```
for i → [0, n-1]
    for j → [0, i]
```

i	j	Iterations
0	0 → 0	1
1	0 → 1	2
2	0 → 2	3
3	0 → 3	4
⋮		
n-1		

Total = $1 + 2 + 3 + 4 + 5 + \dots N$

$$= \frac{N(N+1)}{2}$$

$1 <= N <= 10^3$

$1 <= arr[i] <= 10^5$

$O(N^2)$

$O(\max(arr[i]))$

```
arr = []
for i → [0, n-1]
    arr.append(i)
```

} Space - O(N)

o, Θ, Ω, → Discrete mathematics

```
for (i=0; i<100; i++)
```

N = 100 → 100 iterations

$N = 10^{25}$ → 100 iterations

Does no of iterations depend on N?

↳ No

Constant no of iterations = 100

Time - O(1)

```
main() {   Come from input
    sum(N, M)
}
```

```
sum(a, b) {
    return arr[a+b]
}
```

↓

Space - $O(N+M)$

Good Night

Thank You

Friday