

Week 3 Lecture Notes

Explanation of Operations Cost in List

$$L = [1, 2, 3, 4, 5, 6]$$

$\begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 \end{matrix}$

$$L.\text{pop}(3)$$

$$[1, 2, 3, \quad, 5, 6]$$

$\underbrace{\hspace{1.5cm}}_{n-i-1} \rightarrow \text{pop}$

$$\begin{aligned} n &= 6 \\ i &= 3 \end{aligned}$$

$$6 - 3 - 1 \Rightarrow \underline{2} \text{ shift left ops.}$$
$$\{1, 2, 5, 6\}$$

Total operation:

$$\underbrace{n - i - 1}_{n-i} + \underbrace{1}_1$$

Slicing

$$L = \{1, 2, 3, 4, 5, 6, 7\}$$
$$\text{new-list} = L[a:b] = \begin{matrix} b-a \\ 3 \end{matrix}$$
$$\text{new-list} = [3, 4, 5]$$

$\begin{matrix} / & / & / \\ 1 & 2 & 3 \end{matrix}$

Worst case for Pop:

$$L = [1, 2, 3, 4, 5]$$
$$L = \text{pop}(0)$$

$\underbrace{\hspace{1.5cm}}_{\text{shift left ops}}$

$$\underline{L = [2, 3, 4, 5]}$$

$$\text{new list} = L[:]$$

Explanation of discrepancies in duplicates 1 algorithm

```

duplicates.py
1 def duplicates_1(L):
2     n = len(L)
3     for i in range(n):
4         for j in range(n):
5             if i != j and L[i] == L[j]:
6                 return True
7
8     return False

```

$L = ['A', 'B', 'C', 'D']$

		A	B	C	D
i \ j		0	1	2	3
A	0		•	•	•
B	1	•		•	•
C	2	•	•		•
D	3	•	•	•	

Asymptotic analysis of duplicates 1.py program

```

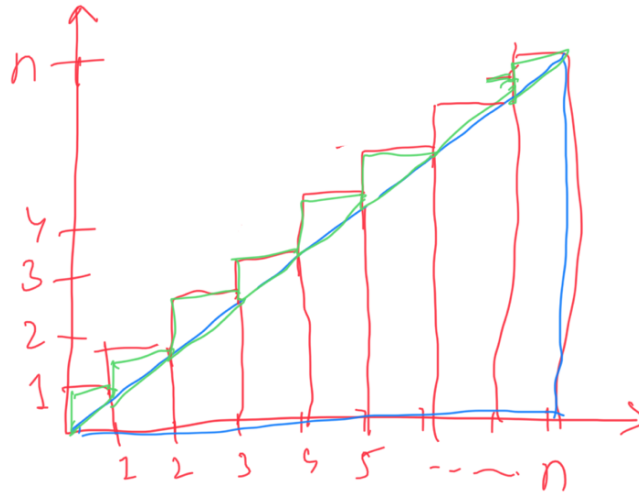
duplicates.py
1 def duplicates_1(L):
2     n = len(L)
3     for i in range(n):
4         for j in range(n):
5             if i != j and L[i] == L[j]:
6                 return True
7
8     return False

```

Handwritten annotations for asymptotic analysis:

- A red circle on the left side of the code, spanning lines 2 through 8, contains the numbers 2, n , n , 2, and 1, representing the number of operations for each line.
- A red circle around the condition `i != j` in line 5.
- Handwritten red text $2 + n(2n) + 1$ and $2n^2 + 3$ at the bottom right.

Sum of K integers identity



∴ Triangle area
 $\frac{1}{2} (\text{width} \times \text{height})$

$$\frac{1}{2} (n \times n)$$

$$\frac{n^2}{2} + n \left(\frac{1}{2} \times 1 \times 1 \right)$$

$$\frac{n^2}{2} + \frac{n}{2}$$

$$\frac{n^2 + n}{2}$$

$$\boxed{\frac{n(n+1)}{2}} \rightarrow \text{Sum of all integers upto } n.$$

$$\frac{3(3+1)}{2} \Rightarrow 6$$

$$\frac{100^5(100+1)}{2} = 5050$$

Asymptotic analysis of duplicates 2

```
10 def duplicates_2(L):
11     n = len(L)
12     for i in range(1,n):
13         for j in range(i):
14             if L[i] == L[j]:
15                 return True
16     return False
```

How many times instruction at L14 runs

$$\underline{1 + 2 + 3 + \dots + n-1}$$

If we have to add all integers to k

$$\frac{k(k+1)}{2} \quad \text{--- (A)}$$

putting $k = n-1$ in (A)

$$\frac{(n-1)(n-1+1)}{2}$$

$$\frac{(n-1)n}{2}$$

$$\frac{n^2}{2} - \frac{n}{2} \rightarrow \text{Atomic operations for L12-14}$$

$$2 + \frac{n^2}{2} - \frac{n}{2} + 1$$

$$\boxed{\frac{n^2}{2} - \frac{n}{2} + 3}$$

Mathematical explanation of Big-O Notation.

The mathematical definition of Big O notation for a given function

$$f(n) = O(g(n)) \quad \text{--- (A)}$$

provided there is a

- constant, $c > 0$

- $n_0 > n$ (threshold value)

$$f(n) \leq c * g(n) \quad \text{--- (B)} \quad \checkmark$$

Example:

$$f(n) = 3n^2 + 2n + 2 \quad \text{--- (1)}$$

I can say that (1) can be represented in big O notation as

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

when

$$f(n) \leq c * n^2 \quad \text{--- (2)}$$

let say $c = 4$

at $n = 1$

$$\begin{aligned} \text{(1)} \Rightarrow f(n) &= 3(1)^2 + 2(1) + 2 \\ &= 7 \end{aligned}$$

$$\begin{aligned} \text{(2)} \Rightarrow 4 \times (1)^2 \\ 4 \end{aligned}$$

at $n = 2$

$$\begin{aligned} \text{(1)} \Rightarrow &= 3(2)^2 + 2(2) + 2 \\ &= 12 + 4 + 2 \\ &= 18 \end{aligned}$$

$$\begin{aligned} \text{(2)} \Rightarrow 4(2)^2 \\ 16 \end{aligned}$$

at $n = 3$

$$\begin{aligned} &= 3(3)^2 + 2(3) + 2 \\ &= 27 + 6 + 2 \\ &= 35 \end{aligned}$$

$$\begin{aligned} \text{(2)} \Rightarrow 4(3)^2 \\ 36 \end{aligned}$$

At $n = 3$, (2) satisfies

