a = [1,2,3,4,5]

ans = [ ]

for i in a: — iteration

ans.append(i ** 2) — exp

[i ** 2 for i in a]

exp — iteration

```
for i in range(0,10):
    for j in range(0,10):
        print (i x j)
```

[print(i x j) for i in range(0,10) for j in range(0,10)

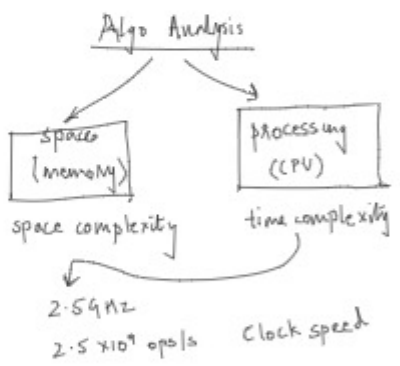
[1, 2, 3, [4, 5]]

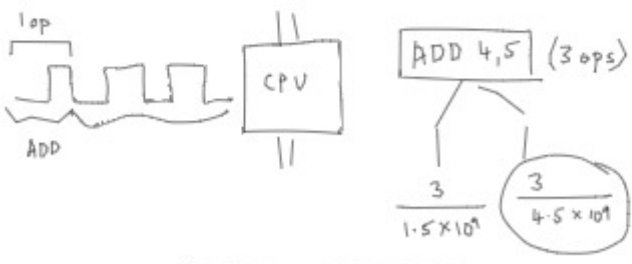
result = [ ]

e = [4, 5]

result.extend([4, 5])

result = [4, 5]





TC → No. of operations

Cost(f(x)) = $x^n + x^{n-1} + \dots + 1$

TC ↑ highest power

Big O time complexity

$O(x^n)$

Big O → O(n)

trend ↑ polynomial term with highest power

$f(x) = x + c$

$f(x) = x^1 + cx$

$O(f(x)) = x^1$

```
def foo(n):                          ——> 1
    for i in range(n):               ——> n
        print(i)
    for i in range(n):               ——> n
        for j in range(n):           ——> n      n * n = n^2
            print(i+j)
```

$C(f(x)) = n^2 + n + 1(c)$

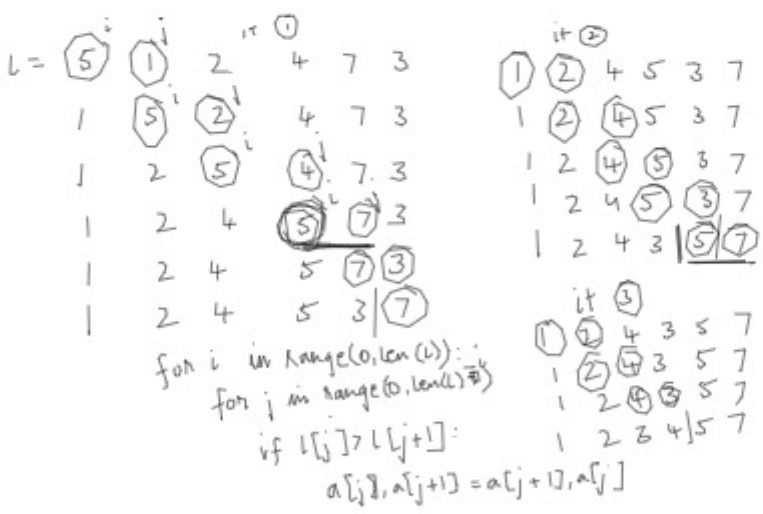
$O(n^2)$

Sorting

Bubble | Selection | Insertion | Merge

TIM SORT

```
for i in range(0, len(L)):
    for j in range(0, len(L)-1):
        if L[j] > L[j+1]:
            a[j], a[j+1] = a[j+1], a[j]
```



$C(f(x)) = n + n-1 + n-2 + \dots$

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

$= \frac{n^2 - n}{2}$

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

i — — — — — — len(i) = 6

i — — — — — ✗