# • Bubble Sort

## Pseudo Code:-

1. for i=0 to A.length → use python technique
2. for J=0 to (A.length-i-1)
   If [A]
3. If A[J] > A[J+1]
4. exchange A[J] with A[J+1]

## Dry Running At Average Case :-

A = [3,5,1,2]

## First Iteration Of For:-

1. i=0

• First iteration of for

2. J=0
3. If 3 > 5
4. _____

· 2nd iteration of for

2. J = 1
3. If 5 > 1
4. swapping → A = [3, 1, 5, 2]

· 3rd iteration of for

2. J = 2
3. If 5 > 2
4. swapping → A = [3, 1, 2, 5]

## SECOND ITERATION OF FOR

1. i = 1
first iteration of for
2. J = 0
3. If 3 > 1
4. swapping → A = [1, 3, 2, 5]

· 2nd iteration of for

~~1. i = 2~~ 2. J = 1
3. If 3 > 2
4. swapping → A = [1, 2, 3, 5]



Scanned with CamScanner

## 3RD ITERATION OF FOR

1. i = 2
   first iteration of for
2. J = 0
3. If 1 > 2
4. _____

## 4TH ITERATION OF FOR

1. i = 3
   first iteration of for
2. ~~J = 0~~ loop terminates = J = 0
3. ~~If 1 > 2~~
4. _____

## FIFTH ITERATION OF FOR

Loop terminates

4 +3 +2 +1
3 +2 +1

# ANALYSIS TABLE

Date

| Line no of code | Time/ instruction | frequency (Average case) | Best case |
|---|---|---|---|
| 1 | c | n+1 | n+1 |
| 2 | c | x | $\sum_{J=1}^{n} J \cdot \frac{n(n+1)}{2}$ |
| 3 | c | x-1 | $\sum_{J=1}^{n} J-1 \cdot \frac{n(n-1)}{2}$ |
| 4 | c | x-1 | 0 |

## T(n) FOR BEST CASE:-

$$T(n) = C\left[n+1 + n\left(\frac{n+1}{2}\right) + n\left(\frac{n-1}{2}\right) + 0\right]$$

$$T(n) = C\left[n+1 + \frac{n^2+n}{2} + \frac{n^2-n}{2}\right]$$

$$T(n) = C\left[\frac{2n+2+n^2+n+n^2-n}{2}\right]$$

$$T(n) = C\left[\frac{2n^2+2n+2}{2}\right]$$

$$T(n) = C(n^2+n+1)$$

# TIME COMPLEXITY
## WORST CASE

| Line no of code | Time/ instruction | frequency (worst case) |
|---|---|---|
| 1 | C | $n+1$ |
| 2 | C | $\sum_{J=1}^{n} J \cdot \frac{n(n+1)}{2}$ |
| 3 | C | $\sum_{J=1}^{n} J-1 \cdot \frac{n(n-1)}{2}$ |
| 4 | C | $\sum_{J=1}^{n} J-1 \cdot \frac{n(n-1)}{2}$ |

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

$$T(n) = \frac{2n+2+n^2+n+n^2-n+n^2-n}{2}$$

$$T(n) = \frac{3n^2+n+2}{3}$$

## DISCUSSION:-

Bubble sort in worst case grows quadratically