1. If $t_1(n) \in O(g_1(n))$ and $t_2(n) \in O(g_2(n))$, prove that $t(n) + t_2(n) \in O(\max\{g_1(n), g_2(n)\})$

For any four arbitrary real number a_1, b_1, a_2, b_2, we have $a_1 + a_2 \leq 2\max\{b_1, b_2\}$

Since $t_1(n) \in O(g_1(n))$, then there exists some constant c_1.

$t_1(n) \leq c_1 g_1(n)$ for all $n \geq n_1$

$t_2(n) \leq c_2 g_2(n)$ for all $n \geq n_2$

Let $c_3 = \max\{c_1, c_2\}$ and $n_0 = \max\{n_1, n_2\}$

$$t_1(n) + t_2(n) \leq c_1 g_1(n) + c_2 g_2(n)$$
$$\leq c_3 g_1(n) + c_3 g_2(n)$$
$$\leq c_3 \{g_1(n) + g_2(n)\}$$
$$\leq 2c_3 \max\{g_1(n), g_2(n)\}.$$

$\therefore$ $t_1(n) + t_2(n) \in O(\max\{g_1(n), g_2(n)\})$, with constants c and no required by the O definition being $2c_3 = 2\max\{c_1, c_2\}$ and $\max\{n_1, n_2\}$.

Find the time complexity of below of the recurrence equation.

$$T(n) = \begin{cases} 2T\left(\frac{n}{2}\right) + 1 & \text{if } n > 1. \\ 1 & \text{otherwise.} \end{cases}$$

$T(n) = aT\left(\frac{n}{b}\right) + f(n)$ Masters Theorem.

$a = 2$

$b = 2$

$\log_b^a = \log_2^2 = 1$

$k = 0$

$\log_b^1 a > 0$

$> k$

case 1)

$\theta(n^{\log_b a})$

$\theta(n \cdot 1)$

$\theta(n)$

4. $T(n) = \begin{cases} 2T(n-1) & \text{if } n>0 \\ 1 & \text{otherwise.} \end{cases}$

Backward substitution:-

$T(n) = 2T(n-1) \rightarrow (1)$ Initial $T(0)=0$

$n = n-1$

$T(n-1) = 2T((n-1)-1)$

$T(n-1) = 2T(n-2) \rightarrow (2)$

Sub (2) in (1)

$T(n) = 2[2T(n-2)]$

$T(n) = 2^2 T(n-2) \rightarrow (3)$

$n = n-2$

$T(n-2) = 2T((n-2)-1)$

$T(n-2) = 2T(n-3) \rightarrow (4)$

Sub (4) in (3)

$T(n) = 2^2[2T(n-3)]$

$T(n) = 2^3 T(n-3) \rightarrow (5)$

$n = n-3$

$T(n-3) = 2T((n-3)-1)$

$T(n-3) = 2T(n-4) \rightarrow (6)$

Sub (6)

BF

sub ⑥ in ⑤

$$T(n) = 2^3 [2T(n-4)]$$

$$= 2^4 T(n-4) \rightarrow ⑦$$

$$T(n) = 2^k T(n-k)$$

$$n-k=0 \Rightarrow n=k$$

if $T(0)=1$

$$T(n) = 2^k \cdot T(0)$$

$$T(n) = 2^k \cdot 1$$

$$T(n) = 2^k$$

$\therefore n = k$

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

5. Big O Notation: Show that $f(n) = n^2+3n+5$ is $O(n^2)$

To Prove that $f(n) = n^2+3n+5$ is $O(n^2)$ we need to find constant c and n_0 such that

$f(n) \leq c \cdot n^2$ for all $n \geq n_0$

$$f(n) = n^2+3n+5$$

For $n \geq 1$, $n^2 \geq n$ ... so on

$$f(n) = n^2+3n+5 \leq n^2+3n^2+5n^2$$

$$f(n) = n^2+3n+5 \leq 9n^2 \text{ for } n \geq 1$$

so, for $c=9$ and $n_0=1$

$f(n) \leq c \cdot n^2$ for all $n \geq n_0$

That Proves $f(n)$ is $O(n^2)$.

6. Big omega Notation : Prove that $g(n) = n^3 + 2n^2 + 4n$ is $\Omega(n^3)$.

To prove that $g(n) = n^3 + 2n^2 + 4n$ is $\Omega(n^3)$

We need to find constants c and n_0 such that

$g(n) \geq c \cdot n^3$ for all $n \geq n_0$

$g(n) = n^3 + 2n^2 + 4n$

For $n \geq 1$

$g(n) = n^3 + 2n^2 + 4n \geq n^3$

Since $2n^2$ and $4n$ are both less than n^3 when $n \geq 1$

So, for $c = 1$ and $n_0 = 1$

$g(n) \geq c \cdot n^3$ for all $n \geq n_0$

That proves $g(n)$ is $\Omega(n^3)$.

7. Big Theta Notation : Determine whether $h(n) = 4n^2 + 3n$ is $\Theta(n^2)$ or not

1. $h(n) = 4n^2 + 3n$ is $O(n^2)$

For $n \geq 1$, $h(n) \leq 4n^2 + 3n^2$

(since $3n$ is less than n^2 when $n \geq 1$)

For this simplifies to $h(n) \leq 7n^2$

for $n \geq 1$

$\therefore$ $h(n)$ is $O(n^2)$.

8. Let
Ω (g
answ
$n = 1$
$n =$
$n =$
n

$n^2 + 4n$

$2.\ h(n) = 4n^2 + 3n$ is $\Omega(n^2)$:

for $n \geq 1$, $h(n) \geq 4n^2$

(since $3n$ is positive)

Therefore $h(n)$ is $\Omega(n^2)$

Since $h(n)$ is both $O(n^2)$ and $\Omega(n^2)$, it is $\theta(n^2)$.

8. Let $f(n) = n^3 - 2n^2 + n$ and $g(n) = -n^2$. show whether $f(n) = \Omega(g(n))$ is true or false and justify your answer.

$n = 1$

$f(1) = 1^3 - 2(1)^2 + 1$

$= 1 - 2 + 1$

$= 0$

$g(1) = -(-n)^2$

$= (-1)^2$

$= 1$

$n = 2$

$f(2) = 2^3 - 2(2)^2 + 2$

$= 8 - 8 + 1$

$= 2$

$g(2) = (-2)^2$

$= 4$

$n = 3$

$f(3) = 3^3 - 2(3)^2 + 3$

$= 27 - 9 + 3$

$= 21$

$g(3) = (-3)^2$

$= 9$

$n = 4$

$f(4) = 4^3 - 2(4)^2 + 4$

$= 64 - 32 + 4$

$= 32 + 4$

$= 36$

$g(n) = (-4)^2$

$= 16$

$n=5$ $\qquad g(n) = (-5)^2$

$= 25$

$f(5) = 5^3 - 2(5)^2 + 5$

$= 15 - 50 + 5$

$= 35 + 5$

$= 40$

$f(n) \leq g(n)$

So it is best case according to asymptotic notation.

$f(n) = \Omega(g(n))$.

9. Determine whether $h(n) = n \log n + n$ is in $\Theta(n \log n)$. Prove a rigorous proof for your conclusion.

1. Upper Bound (O notation):-

we need to find c_1 and n_0 such that

$h(n) \leq c_1 \cdot n \log n$ for all $n \geq n_0$

$h(n) = n \log n + n$

$\leq n \log n + n \log n$ (since $\log n$ is increasing)

$= 2n \log n$.

Now, Let $c_1 = 2$, then $h(n) \leq 2n \log n$ for all $n \geq 1$.

So, $h(n)$ is $O(n \log n)$.

2. Lower bound (Ω notation):-

we need to find c_2 and n_0 such that

$h(n) \geq c_2 \cdot n \log n$ for all $n \geq n_0$

$h(n) = n \log n + n$

Solve

ord

$\geq \frac{1}{2} \cdot n \log n$ (for $n \geq 2$)

Now, let $c_2 = \frac{1}{2}$, then $h(n) \geq \frac{1}{2} \cdot n \log n$ for all $n \geq 2$. So, $h(n)$ is $\Omega(n \log n)$

3. Combining bounds :-

Since $h(n)$ is both $O(n \log n)$ and $\Omega(n \log n)$, it ~~is~~ also $\Theta(n \log n)$

Thus, $h(n) = n \log n + n$ is in $\Theta(n \log n)$.

10. Solve the following recurrence relations and find the order of growth for solution.

$T(n) = 4T(n/2) + n^2$, $T(1) = 1$.

$T(n) = aT(n/b) + f(n)$

$a = 4$
$b = 2$

$\log_b^a = \log_2^4 = 2$

$k = 2$

$2 = 2$

$\log_b^a = k$

$p > -1 \quad \Theta(n^k \log_n^{p+1})$

$\Theta(n^2 \cdot \log_n^{1+1})$

$\Theta(n^2 \cdot \log_n^2)$

$T(n) = \Theta(n^2 \cdot \log(n))$

The order of growth for the solution is $n^2 \cdot \log(n)$.

11. Given an array of {4, -2, 5, 3, 10, -5, 2, 8, -3, 6, 7, -4, 1, 9, -1, 0, -6, -8, 11, -9} integers, Find maximum and Minimum product that can be obtained by multiplying 2 integers from array.

Given :- [4, -2, 5, 3, 10, -5, 2, 8, -3, 6, 7, -4, 1, 9, -1, 0, -6, 11, -9].

Maximum product:-

2 Largest no's : 11, 10

2 Smallest (-ve no's) : -9, -8

Products :-

$11 \times 10 = 110$

$-9 \times -8 = +72$

$\therefore$ Max product $= 110$.

Minimum Product :-

$11 \times -9 = -99$

$10 \times -9 = -90$

$\therefore$ Mini Product $= -99$

12. Demonstrate Binary search method to search key = 23 from the array arr[] = [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

arr[] = [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Key = 23.

0	1	2	3	4	5	6	7	8	9
2	5	8	12	16	23	38	56	72	91

$M = \frac{L+h}{2}$

3). Apply

Data

recu

Made

$M = \frac{l+h}{2} = \frac{0+9}{2} = 4.5 \simeq 5$

0	1	2	3	4	5	6	7	8	9
2	5	8	12	16	$\boxed{23}$ Mid	38	56	72	91

$\therefore$ arr [Mid] = 23

arr [Mid] = key

23 = 23

$\therefore$ key is found.

3) Apply merge sort and order the list of 8 element Data d = (45, 67, -12, 5, 22, 30, 50, 20). Set up a recurrence relation for the number of key comparison made by Mergesort.

d = 45, 67, -12, 5, 22, 30, 50, 20 (indices 0 to 7)

$m = \frac{0+7}{2} = 4$

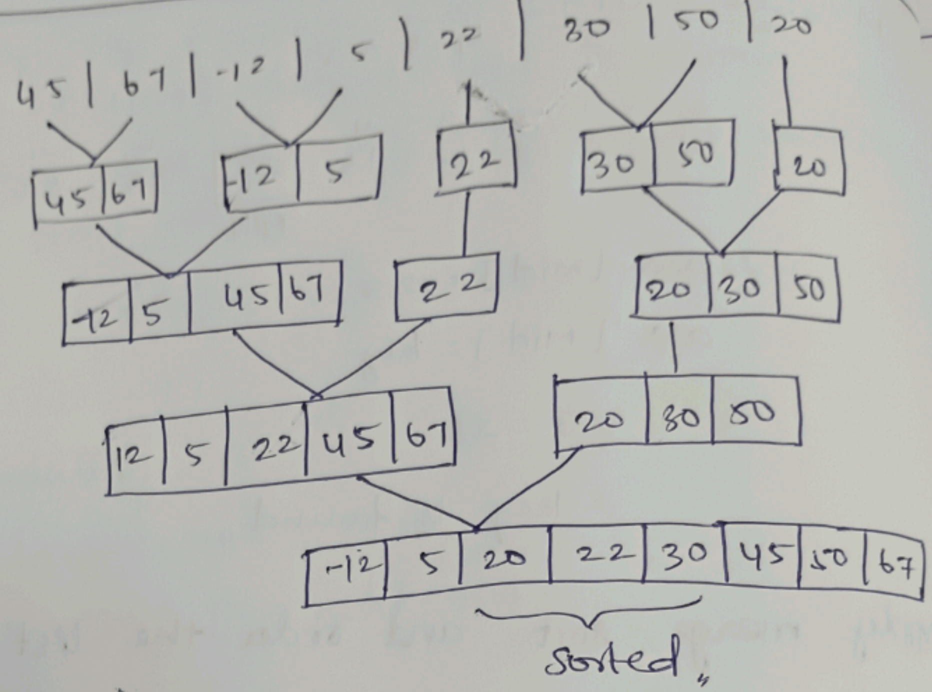
0	1	2	3	4		5	6	7
45	67	-12	5	22	|	30	50	20

$M = \frac{0+4}{2} = 0$

0	1	2		3	4		5	6		7
45	67	-12	|	5	22	|	30	50	|	20

$M = \frac{0+2}{2} = 1$

0	1		2		3		4		5		6		7
45	67	|	-12	|	5	|	22	|	30	|	50	|	20



Recurrence Relation:-

$$T(n) = 2T\left(\frac{n}{2}\right) + c(n)$$

$a = 2 \quad , \quad k = 1$

$b = 2 \quad , \quad p = 1$

$\log_b^a = \log_2^2 = 1$

$\Rightarrow \log_b^a = k$

$\therefore \Theta(n^k \log_n^{p+n})$

$\Theta(n^1 \log_n^2)$

$\therefore \Theta(n \log n)$.

14. Find the no of times to perform swap selection sort. Also estimate the time complexity the order of notation sets (12, 7, 5, -2, 18, 6, 13, 4)

S = 12, 07, 5, -2, 18, 6, 13, 14.

Step 1 :- 12 7 5 -2 18 6 13 14

Start ↓ 12 ; Mini ↓ -2

Step 2 :- -2 7 5 12 18 6 13 14

Start ↓ 7 ; Min ↓ 5

Step 3 :- -2 5 7 12 18 6 13 14

Start ↓ 7 ; Min ↓ 6

Step 4 :- -2 5 6 7 18 12 13 14

Start ↓ 18 ; Min ↓ 12

Step 5 :- -2 5 6 7 12 18 13 14

Start ↓ 18 ; Min ↓ 13

Step 6 :- -2 5 6 7 12 13 18 14

Start ↓ 18 ; Min ↓ 14

Step 7 :- $\underbrace{-2 \quad 5 \quad 6 \quad 7 \quad 12 \quad 13}_{\text{sorted}}$ 14 18.

Time complexity

Best - $O(n^2)$

Avg - $O(n^2)$ Space complexity = $O(1)$

Worst - $O(n^2)$

Total No of swaps = 6.

15. Find the index of the target value 10 using bina Search from the following list of elements.

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20].

0	1	2	3	4	5	6	7	8	9
2,	4	6	8	10	12	14	16	18	20

$M = \frac{l+h}{2} = \frac{0+9}{2} = 4.5 \approx 5 (Or) 4.$

0	1	2	3	4	5	6	7	8	9
2	4	6	8	10	12	14	16	18	20

Target = 10

a [mid] = Target

∴ 10 = 10

∴ Target found.

16. Sort and 60

M

n

m=

16. Sort the following elements using merge sort divide and conquer strategy [38, 27, 43, 3, 9, 82, 10, 15, 88, 52, 60, 5] and analyze complexity of algorithm.

0	1	2	3	4	5	6	7				
38	27	43	3	9	82	10	15	88	52	60	5

$M = \frac{l+h}{2} = \frac{0+11}{2} = 5.5 \approx 6$

0 1 2 3 4 5 6 7 8 9 10 11

38 27 43 3 9 82 10 | 15 88 52 60 5

$m = \frac{l+h}{2} = \frac{0+6}{2} = 3$

0 1 2 3 4 5 6 7 8 9 10 11

38 27 43 3 | 9 82 10 | 15 88 52 | 60 5

$m = \frac{l+h}{2} = \frac{0+3}{2} = 2$

$m = \frac{l+h}{2} = \frac{4+6}{2} = 5$

$M = \frac{7+9}{2} = 8$

$M = \frac{10+11}{2} = \frac{21}{2} = 10$

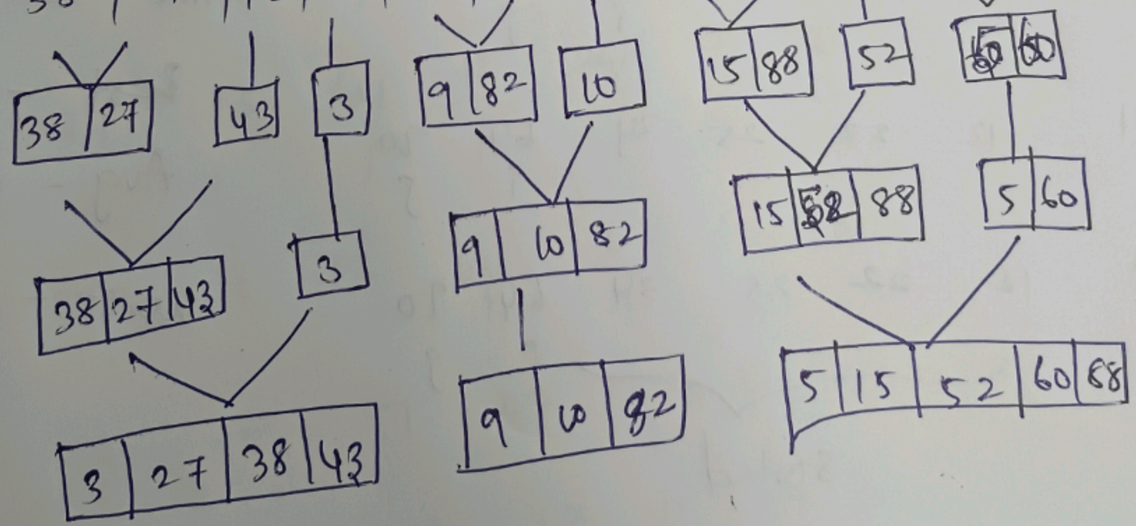
0 1 2 3 4 5 6 7 8 9 10 11

38 27 | 43 | 3 | 9 | 82 | 10 | 15 | 88 | 52 | 60 | 5

M = 0

0 1 2 3 4 5 6 7 8 9 10 11

38 | 27 | 43 | 3 | 9 | 82 | 10 | 15 | 88 | 52 | 60 | 5



It-4

12 22 11 25 34 64 90
i j

12 22 11 25 34 64 90
i j

12 11 22 25 34 64 90
i j

12 11 22 25 34 64 90
i j

12 11 22 25 34 64 90
i j

12 11 22 25 34 64 90
i j

It-5

12 11 22 25 34 64 90
i j

11 12 22 25 34 64 90
i j

11 12 22 25 34 64 90
i j

11 12 22 25 34 64 90
i j

11 12 22 25 34 64 90
i j

$\underbrace{11 \quad 12 \quad 22 \quad 25 \quad 34}_{\text{sorted}} \quad 64 \quad 90$
i j

Time complexity

Best - $O(n)$

Avg - $O(n^2)$

Worst - $O(n^2)$.

3	9	27	38	43	82

5	15	52	60	88

3	9	27	38	43	82	5	15	52	60	88

3	5	9	15	27	38	43	52	60	82	88

= sorted.

Time complexity :-

Best = $O(n^2)$

Avg = $O(n^2)$

worst = $O(n^2)$.

17). Sort the array 64, 34, 25, 12, 22, 11, 90 using Bubble sort. what is time complexity of Bubble sort in best, worst, average cases!

Given.

64 34 25 12 22 11 90
i j

34 64 25 12 22 11 90
i j

34 25 64 12 22 11 90
i j

34 25 12 64 22 11 90
i j

34 25 17 22 64 11 90
i j

34 25 12 22 11 64 90
i j

34 25 12 22 64 11 90
i j

34 25 12 22 11 64 90
i j

It-2

34 25 12 22 11 64 90
i j

25 34 12 22 11 64 90
i j

25 12 34 22 11 64 90
i j

25 12 22 34 11 64 90
i j

25 12 22 11 34 64 90
i j

It-3

25 12 22 11 34 64 90
i j

12 25 22 11 34 64 90
i j

12 22 25 11 34 64 90
i j

12 22 11 25 34 64 90
i j

12 22 11 25 34 64 90
i j

12 22 11 25 34 64 90
i j

18) Sort

Sort.

the

Give

18) Sort the array 64, 25, 12, 22, 11 using selection sort. What is time complexity of selection sort in the best, worst, average cases.

Given:-

0	1	2	3	4
64	25	12	22	11

start — min

11	25	12	22	64

sorted — start — min

11	12	25	22	64

sorted — start — min

11	12	25	25	64

sorted.

Time complexity:-

Best case : $O(n^2)$

Avg case : $O(n^2)$

Worst case : $O(n^2)$

19. Sort the following elements using Insertion sort using Brute Force approach strategy [38, 27, 43, 3, 9, 82, 10, 15, 88, 52, 60, 5] and analyze complexity algorithm.

Given:-

[38 27 43 3 9 82 10 15 88 52 60 5]

1) 27 38 43 3 9 82 10 15 88 52 60 5

2) 27 38 43 3 9 82 10 15 88 52 60 5

3) 27 38 3 43 9 82 10 15 88 52 60 5

4). 3 27 38 43 9 82 10 15 88 52 60 5

5) 3 9 27 38 43 82 10 15 88 52 60 5

6). 3 9 10 27 38 43 82 15 88 52 60 5

7). 3 9 10 15 27 38 43 82 88 52 60 5

8). 3 9 10 15 27 38 43 82 88 52 60 5

9). 3 9 10 15 27 38 43 52 82 88 60 5

10). 3 9 10 15 27 27 38 43 52 82 60 8

11). 3 9 10 15 27 38 43 52 60 82 88 5

12). 3 9 10 15 27 38 43 52 60 82 88

13). 3 5 9 10 15 27 38 43 52 60 82 8

$\underbrace{\hspace{10em}}_{\text{sorted}}$

Time Complexity:-

Best case - O(n) - This occurs when the ar is already sorted. The inner loop will run only once

Avg case - $O(n^2)$ - The list is randomly ordered.

Worst case - $O(n^2)$ - If the list is in reverse.

Space complexity:-

$$O(1) = \text{Insertion sort.}$$

Given an array of [4, -2, 5, 3, 10, -5, 2, 8, -3, 6, 7, -4, 1, 9, -1, 0, -6, -8, 11, -9] integers, sort the following elements using Insertion sort using Brute force approach strategy analyze complexity of algorithm.

4 -2 5 3 10 -5 2 8 -3 6 7 -4 1 9 -1 0 -6 -8 11 -9
i j

-2 4 5 3 10 -5 2 8 -3 6 7 -4 1 9 -1 0 -6 -8 11 -9
i j

-2 4 5 3 10 -5 2 8 -3 6 7 -4 1 9 -1 0 -6 -8 11 -9
i j

-2 4 3 5 10 -5 2 8 -3 6 7 -4 1 9 -1 0 -6 -8 11
i j

-2 3 4 5 10 -5 2 8 -3 6 7 -4 1 9 -1 0 -6 -8 1
i j

-2 3 4 5 -5 10 2 8 3 6 7 -4 1 9 -1 0 -6 -8 1
i j

-4 1 1 -1 0 -6 -8

-5 -2 3 4 5 2 10 8 -3 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -2 3 4 2 5 10 8 -3 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -2 3 2 4 5 10 8 -3 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -2 2 3 4 5 10 8 -3 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -2 2 3 4 5 8 10 -3 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -2 2 3 4 5 8 -3 10 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -2 2 3 4 5 -3 8 10 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -2 2 3 4 -3 5 8 10 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -2 2 3 -3 4 5 8 10 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -2 2 -3 3 4 5 8 10 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -2 -3 2 3 4 5 8 10 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -3 -2 2 3 4 5 8 10 6 7 -4 1 9 -1 0 -6 -8
i j

-5 -3 -2 2 3 4 5 8 6 10 7 -4 1 9 -1 0 -6 -8
i j

-5 -3 -2 2 3 4 5 6 8 10 7 -4 1 9 -1 0 -6 -8
i j

-5 -3 -2 2 3 4 5 6 8 10 7 -4 1 9 -1 0 -6 -8
i j

-5 -3 -2 2 3 4 5 6 8 7 10 -4 -1 9 -1 0 -6 -8
i j

-5 -3 -2 2 3 4 5 6 7 8 10 -4 -1 9 -1 0 -6 -8
i j

-4
-5 -4
-5 -4
-5 -4
-5 -4
-6 -5
-8 -6
-8 -
-9 -
Time
Best
alrea
once
Av
We

-4 1 9 -10 -6 -8 11 -5
4 1 7 -10 -6 -8 11 -5
1 9 -1 0 -6 -8 11 -5
1 9 -1 0 -6 -8 11 -5
9 -10 -6 -8 11 -5
-10 -6 -8 11 -9
-1 0 -6 -8 11 -8
-1 0 -6 -8 11 -8
-1 0 -6 -8 11
0 -6 -8 11
0 -6 -8
0 -6 -8 11
0 -6 -8 11
0 -6 -8 11
0 -6 -8 11
-6 -8 11
-6 -8 11

-5 -4 -3 -2 2 3 4 5 6 7 8 10

-5 -4 -3 -2 1 2 3 4 5 6 7 8 10 9 -1 0 -6 -8 11 -9
i j

-5 -4 -3 -2 1 2 3 4 5 6 7 8 10 9 -1 0 -6 -8 11 -9
i j

-5 -4 -3 -2 -1 1 2 3 4 5 6 7 8 9 10 -1 0 -6 -8 11 -9
i j

-4 -3 -2 -1 1 2 3 4 5 6 7 8 9 10 0 -6 -8 11 -9
i j

-5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 -6 -8 11 -9
i j

-6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 -8 11 -9
i j

-8 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 ~~11~~ -9
i j

-8 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 11 -9
i j

-9 -8 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 11

Time complexity

is sorted

Best case (O(n)) - This occurs when the array is already sorted. The inner loop will run only once for each element.

Average case $O(n^2)$ - The List is randomly

Worst case $O(n^2)$ - If the list is in reverse order.