ASSIGNMENT E·ROSHINI - 192321167

# CSA 0672- DESIGN AND ANALYSIS OF ALGORITHM FOR POLYNOMIAL PROBLEMS.

1 If $t_1(n) \in O(g_1(n))$ and $t_2(n) \in O(g^2(n))$, Prove that $t_1(n) + t_2(n) \in O(\max\{g_1(n), g_2(n)\})$

for any four arbitrary real numbers $a_1, b_1, a_2, b_2$.

Such that $a_1 \leq b_1$ and $a_2 \leq b_2$.

we have $a_1 + a_2 \leq 2 \max\{b_1, b_2\}$

Since $t_1(n) + O(g_1(n))$ then there exists some constant $c_1$ and non negative integer $n_1$ such that

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

Since $t_2(n) \in O(g_2(n))$, then there exists some constant $c_2$ and non-negative. Integer $n_2$ such that.

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

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

$$\begin{aligned} t_1(n) + t_2(n) &\leq c_1 g_1(n) + c_2 g_2(n) \\ &\leq c_3 g_1(n) + c_3 g_2(n) \\ &= c_3\{g_1(n) + g_2(n)\} \\ &\leq 2c_3 \max\{g_1(n), g_2(n)\} \end{aligned}$$

Hence $t_1(n) + t_2(n) \in O(\max)\{g_1(n), g_2(n)\}$, with constants $c$ and $n_0$ required by the $O$ definition being $2c_3 = 2$ $\max\{c_1, c_2\}$ and $\max\{n_1, n_2\}$ respectively

2. Find the time complexity of the below recurrence equation.

3.
$$T(n) = \begin{cases} 2T\left(\frac{n}{2}\right)+1 & \text{if } n > 1 \\ & \text{otherwise} \end{cases}$$

Masters theorem

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

$a = 2$
$b = 2$

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

$k = 0$ $\quad \log_b^1 a \quad > 0$ $\quad > k$

case i) $\theta(n - \log_b^a)$

$\theta(n - 1)$

$\theta(n)$

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

Backward solution

Initial $T(0) = 0$

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

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

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

sub (2) in (1) $\quad T(n) = 2T((n-2)-1)$

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

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

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

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) 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 $T(0) = 1$

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

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

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

$$n = k$$

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

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

To prove that $f(n) = n^2 + 3n + 5$ if $O(n^2)$ we need to find constant c and no such that

$f(n) \leq c \cdot n^2$ for all $n \geq no$

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

for $n \geq 1$, $n^2 \geq n \ldots$ 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 $no = 1$

$f(n) \leq c \cdot n^2$ for all $n \geq no$

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

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

To prove that $g(n) = n^3 + 2n^2 + 4n$ is $\Omega(n^3)$ we need to find costants c and no 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 $O(n^2)$

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 $\Omega(n^2)$ ~~if~~ 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.

$n = 1$

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

$= 1 - 2 + 1$

$= 0$

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

$- (-1)^2$

$= 1$

$n = 2$

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

$= 8 - 8 + 1$

$= 2$

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

$= 4$

$n = 3$

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

$= 27 - 9 + 3$

$= 21$

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

$= 9$

$n = 4$

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

$= 64 - 32 + 4$

$= 32 + 4$

$= 36$

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

$= 16$

$n = 5$

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

$= 15 - 50 + 5$

$= 35 + 5$

$= 40$

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

$= 25$

$f(n) \geq g(n)$

so it is best case according to asymptotic notation.

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

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

1. upper Bound (O notation):

we need to find $c_1$ and $n_0$ such that.

$h(n) \le c_1 \cdot n\log n$ for all $n \ge n_0$

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

$$\le n\log n + n\log n \quad (\text{since } \log n \text{ is increasing})$$

$$= 2n\log n.$$

Now, let $c_1 = 2$, then $h(n) \le 2n\log n$ for all $n \ge 1$

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

2. Lower Bound ($\Omega$ notation):

we need to find $c_2$ and $n_0$ such that

$h(n) \ge c_2 \cdot n\log n$ for all $n_0 \ge 0$

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

$$\ge \frac{1}{2} \cdot n\log n \quad (\text{for } n \ge 2)$$

now let $c_2 = \frac{1}{2}$, then $h(n) \ge \frac{1}{2} \cdot n\log n$.

for all $n \ge 2$. 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 $\Theta(n\log n)$.

10. solve the following recurrence relations and find the order of growth for solutions.

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

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

$a = 4$
$b = 2$

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

$$k = 2$$

$$p = 2$$

$$\log_b^a = k$$

case ii) $p > -1$ $\theta(n^k \log_n^{p+1})$

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

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

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

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

12. Demonstrate the Binary Search Method to search Key = 23 from the array arr = [] = {2, 5, 8, 12, 16, 23, 38, 56, 72, 91}.

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

low = 0
high = 9

mid = $\frac{low + high}{2}$

$= \frac{0+9}{2} = 4$

a[mid]= =key

a[4] = 23

16! =23

16<23

low= mid+1

| | | | | |
|---|---|---|---|---|
| 23 | 38 | 56 | 72 | 91 |

low= 5 high=9

a[mid]==key

mid = $\frac{5+9}{2} = \frac{14}{2} = 7$

a[7] == 23

56!= 23 56>23

high=mid-1

| | | |
|---|---|---|
| 23 | 38 | 56 |

low=5 high=7

a[6] == 23

mid = $\frac{5+7}{2} = 6$

38 != 23

38>23

high= mid-1

| |
|---|
| 23 |

low=5 high=5

a[5]= =key

23 ==23

mid= $\frac{5+5}{2} = 5$

Return the position of the key (i.e) 5

Procedure :

binary-search (a,n,key):

low = 0

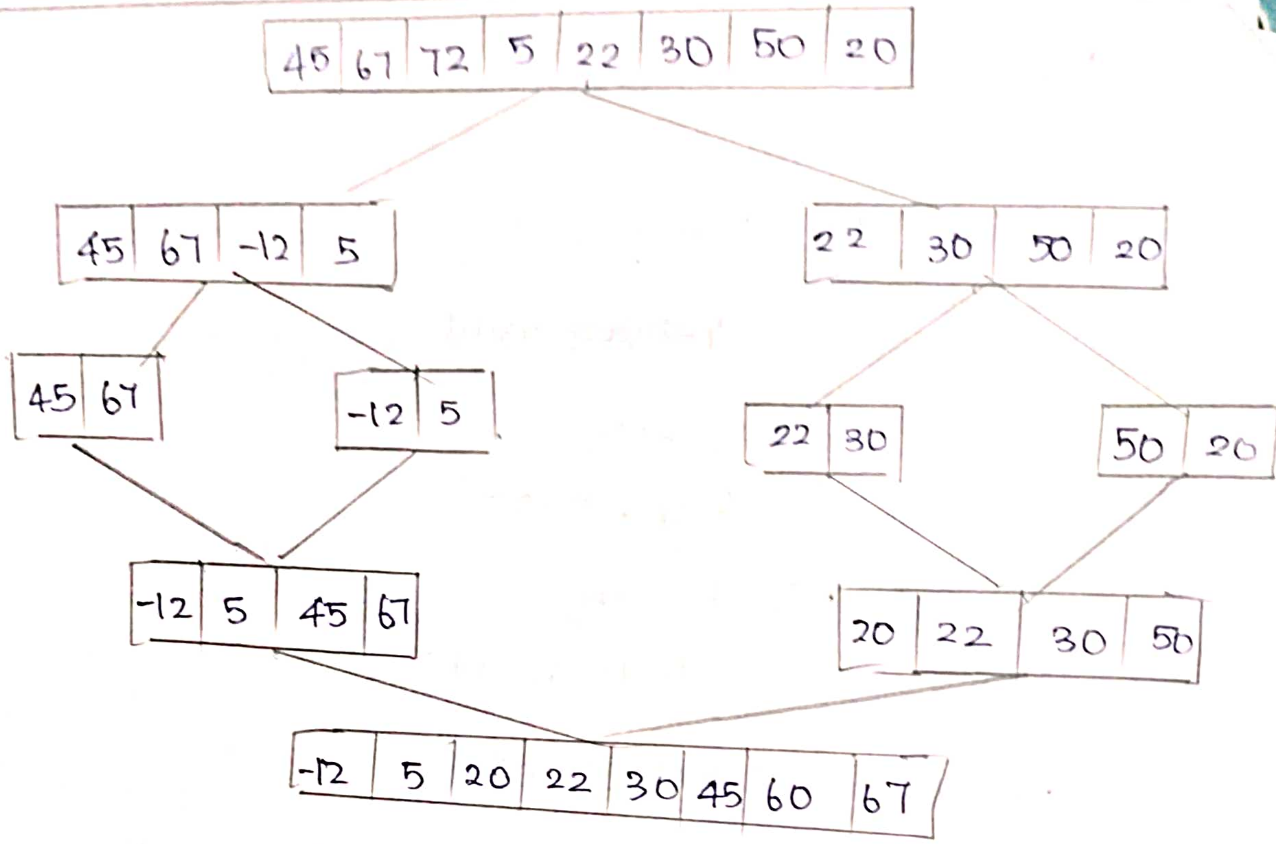
high = n-1

```
while (low <= high):
        mid = (high + low)
        if a[mid] == key
            return mid

       a[mid] > key
            high = mid - 1
       a[mid] < key
            low = mid + 1

   return -1 [if not found]
```

Time complexity :

O(n.log n.

13 Apply merge sort and order the list of 8 elements
d= (45, 67, -12, 5, 22, 30, 50, 20] set up a recurrence
relation for the number of key comparisons made
my merge sort.




∴ The sorted list is :

−12, 5, 20, 22; 30, 45, 60, 67

Time complexity : $O(n \log n)$

Recurrence relation : $T(n) = 2T(n/2) + (n-1)$

14. Find the no. of times to perform swapping for Selection Sort. Also estimate the time complexity.

$S = \{12, 7, 5, -2, 18, 6, 13, 4\}$.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 12 | 7 | 5 | -2 | 18 | 6 | 13 | 4 |

↑ start ↑ min

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -2 | 7 | 5 | 12 | 18 | 6 | 13 | 4 |

↑ start ↑ min

min

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -2 | 4 | 5 | 12 | 18 | 6 | 13 | 7 |

↑ start

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -2 | 4 | 5 | 12 | 18 | 6 | 13 | 7 |

↑ start ↑ min

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -2 | 4 | 5 | 6 | 18 | 12 | 13 | 7 |

↑ start ↑ min

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -2 | 4 | 5 | 6 | 7 | 12 | 13 | 18 |

↓ min

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -2 | 4 | 5 | 6 | 7 | 12 | 13 | 18 |

↑ Start

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -2 | 4 | 5 | 6 | 7 | 12 | 13 | 18 |

↓ min.

sorted list : -2, 4, 5, 6, 7, 12, 13, 18

usally the number of swaps required will be n-1, But for this question there are only 4 swaps.

time complexity : $O(n^2)$. It is $n^2$ in all three cases.

15. Find the index of the target value 10 using binary 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 |

low=0 high=9

mid= $\frac{0+9}{2}$ = 4

a [mid] == key

a [4] = = 10

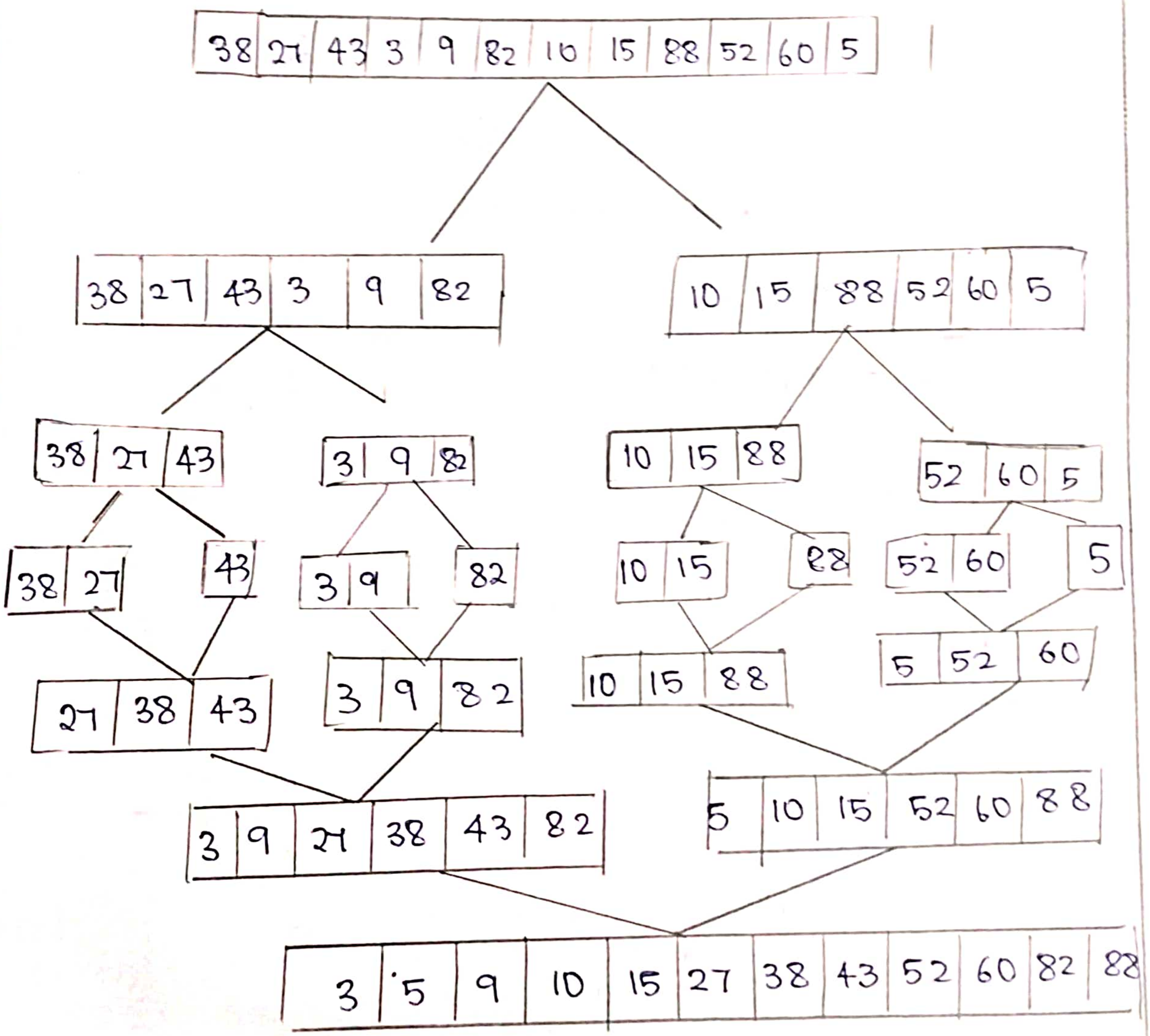
10 = = 10

Return the position of the key (i.e) 4

Pseudocode :

```
binary-search(array, size of array, key)
        low = 0
        high = size - 1
        while (low <= high)
            mid = (high + low)/2
            if a [mid] = = key
                return mid
            a [mid] > key
                high = mid - 1
            a [mid] < key
                low = mid + 1
        return -1 [if not found].
```

16) Solve the elements using Merge sort divide Conquer Strategy [38, 27, 43, 3, 9, 82, 10, 15, 88, 52, 60, 5] & analyze the time complexity




∴ The sorted list is : 3, 5, 9, 10, 15, 27, 38, 43, 52, 60, 82, 88

Pseudocode :

```
Partition (low, high)
    if l<h :
        mid = h+l/2
        Partition (l, mid)
        Partition (mid+1, h)
        merge (l, mid, h)
    end if
```

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

By using Master theorem.

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

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

comparing $\log_a^b$ & $k$

$\log_a^b = k$

$\therefore$ case (ii)

$P < -1$

$\therefore O(n^k \log^{p+1} n)$

$= O(n \log n)$

$\therefore$ Time complexity : $O(n \log n)$

Sort the array 64, 34, 25, 12, 22, 11, 90 using Bubble sort, What is the time complexity of selection Sort in Best, average (worst case).

64 34 25 12 22 11 90

34 64 25 12 22 11 90

34 25 64 12 22 11 90

34 25 12 64 22 11 90

34 25 12 22 64 11 90

34 25 12 22 11 64 90

34 25 12 22 11 64 90

---

34 25 12 22 11 64 90

25 34 12 22 11 64 90

25 12 34 22 11 64 90

25 12 22 34 11 64 90

25 12 22 11 34 64 90

25 12 22 11 34 64 90

---

25 12 22 11 34 64 90

12 25 22 11 34 64 90

12 22 25 11 34 64 90

12 22 11 25 34 64 90

12 22 11 25 $\boxed{34}$ $\boxed{64}$ $\boxed{90}$

---

$\boxed{12 \quad 22}$ 11 25 34 64 90

12 $\boxed{22 \quad 11}$ 25 34 64 90

12 11 $\boxed{22 \quad 25}$ 34 64 90

12 11 22 $\boxed{25}$ $\boxed{34}$ $\boxed{64}$ $\boxed{90}$

$\boxed{12 \quad 11}$ 22 25 34 64 90

11 $\boxed{12 \quad 22}$ 25 34 64 90

11 12 $\boxed{22}$ $\boxed{25}$ $\boxed{34}$ $\boxed{64}$ $\boxed{90}$

---

$\boxed{11 \quad 12}$ 22 25 34 64 90

$\boxed{11}$ $\boxed{12}$ $\boxed{22}$ $\boxed{25}$ $\boxed{34}$ $\boxed{64}$ $\boxed{90}$

∴ The sorted list is : 11, 12, 22, 25, 34, 64, 90

time complexity ( $O(n^2)$

Selection sort : Best case : $O(n^2)$

worst case : $O(n^2)$

Average case : $O(n^2)$

18 Sort the array 64, 25, 12, 22, 11 using selection sort what is the time complexity.

| | | | | | |
|---|---|---|---|---|---|
| 64 | 25 | 12 | 22 | 11 | 90 |

↑start ↑min

| | | | | | |
|---|---|---|---|---|---|
| 11 | 25 | 12 | 22 | 64 | 90 |

↑start ↑min

| | | | | | |
|---|---|---|---|---|---|
| 11 | 22 | 25 | 22 | 64 | 90 |

↑start ↑min

| | | | | | |
|---|---|---|---|---|---|
| 11 | 12 | 22 | 25 | 64 | 90 |

↓ start min

| | | | | | |
|---|---|---|---|---|---|
| 11 | 12 | 22 | 25 | 64 | 90 |

↑start

| | | | | | |
|---|---|---|---|---|---|
| 11 | 12 | 22 | 25 | 64 | 90 |

Time complexity:

$O(n^2)$.

The outer loop runs n+1 times and the inner loops are n-1, n-2, till time.

Best case : $O(n^2)$

Worst case: $O(n^2)$

Average case : $O(n^2)$

19 solve the following using Insertion sort using Brute-force
[38, 27, 43, 3, 9, 82, 10, 15, 18, 52, 60, 5]

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

```
def ins (all):
    n= len(all)
    if n <= 1 :
        return
    for i in range (1,n)
        k = all[i]
        j = i-1
        while j> =0 and k < all [j]:
            all [j+1]= all[j]
            j -=1
        all [j+1] = k
all = [38, 27, 43, 3, 9, 82, 10, 15, 88, 52, 60, 5]
ins(all)
Print(all)
```

E.ROSHINI - 1428...

10 Insertion sort :

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

```
def Ins(arr):
    n = len(arr)
    if n <= 1;
        return
    for i in range(1, n)
        k = arr[i]
        j = i-1
        while j >= 0 and k < arr[j]:
            arr[j+1] = arr[j]
            j -= 1
        arr[j+1] = k
```

arr = [4, -2, 5, 3, 10, -5, 2, 8, -3, 6, 7, -4, 1, 9, -1, 0 -6, -8, 11, -9]

11 Give an array of [4, -2, 5, 3, 10, -5, 2, 8, -3, 6, 7, -4 1, 9, -1, 0, -6, 11 -9] integer, find the maximum and minimum product that can be obtained by multiplying two integers from the array.

```
def Min(all,n):
        Yes =all [0]
        for i  in range (1,n):
                res = min ( reci, ari [i] )
        return Yes.
def Max (all, n) :
        res = arr [0]
        for i  in range (1, n) :
            res = Max (ress - all [i] )
def  product  (all, n)
        Min = get min(arr, n)
        Max  = get Max (all,n)
        return min * Max
all = [4,-2, 5, 3,10, -5,2,8,-3,6,7, -4,1,9,0,6,11,-9]
n = len(all)
Print ("Product"-, Product (all, n)).
```