Quick Select

Sunday, 2 May 2021 9:58 AM

arr → {80 10 20 90 70 60 40 30 50}

0	1	2	3	4	5	6	7	8

find kth Smallest?

K = 3

k-1

partition Index→

quickSelect

correct position

k-1 index

```
pivot = arr[hi];
pi = partitionIndex(arr, lo, hi)
 sorted position
if(pi == k){
    answer is → pivot
} else if(pi > k){
    answer is on left side.
} else if(pi < k){
    answer is on Right side
}
```

k=

Brute force ① Sort Array]

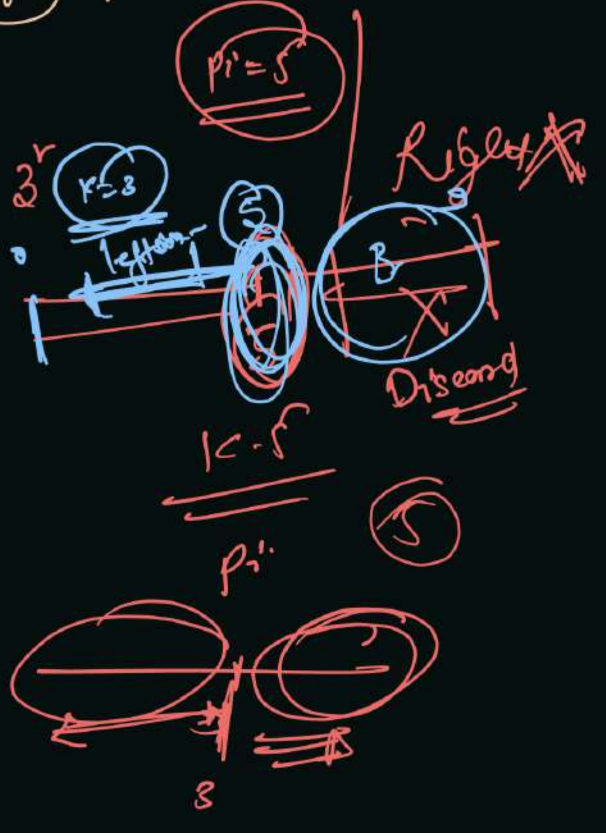
[nlogn]

10 20 30 40 50 60 70 80 90

k=3

think in more optimised Approach.

nlogk Heap not allowed X



0 1 2 3 4 5 6 7

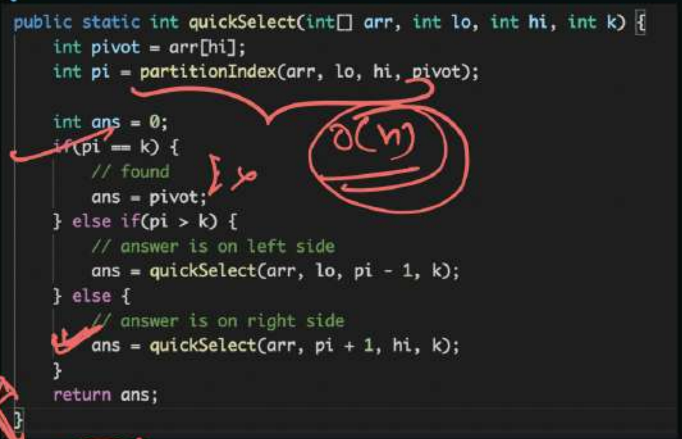
arr → { 90 20 10 70 50 40 60 30 }

↑ lo ↑ hi

Call from main →

quickSelect (arr, lo, hi, k-1)

Index



```
public static int quickSelect(int[] arr, int lo, int hi, int k) {
    int pivot = arr[hi];
    int pi = partitionIndex(arr, lo, hi, pivot);

    int ans = 0;
    if(pi == k) {
        // found
        ans = pivot;
    } else if(pi > k) {
        // answer is on left side
        ans = quickSelect(arr, lo, pi - 1, k);
    } else {
        // answer is on right side
        ans = quickSelect(arr, pi + 1, hi, k);
    }
    return ans;
}
```

O(n)

pivot = ~~30~~ 60

pi = ~~3~~ 5

pi < k

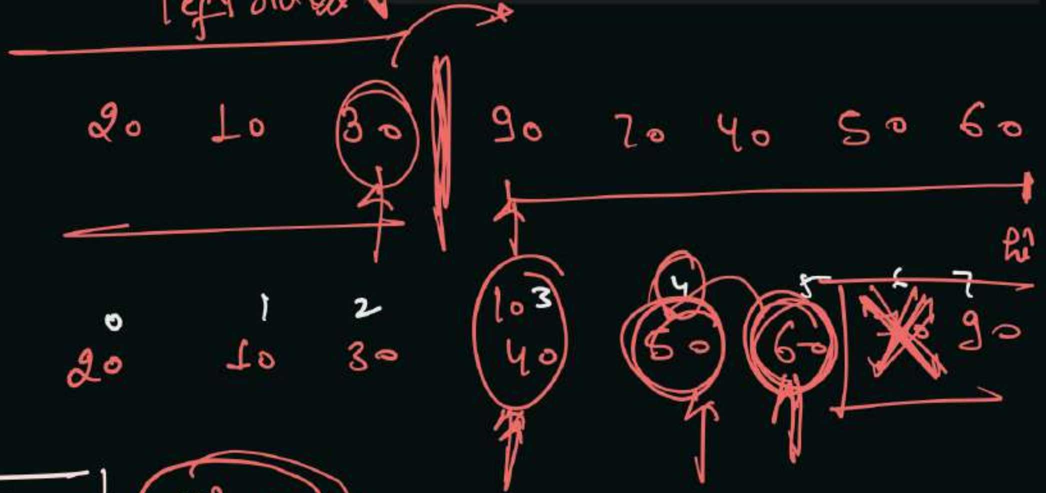
if

pi = 5

pi > k

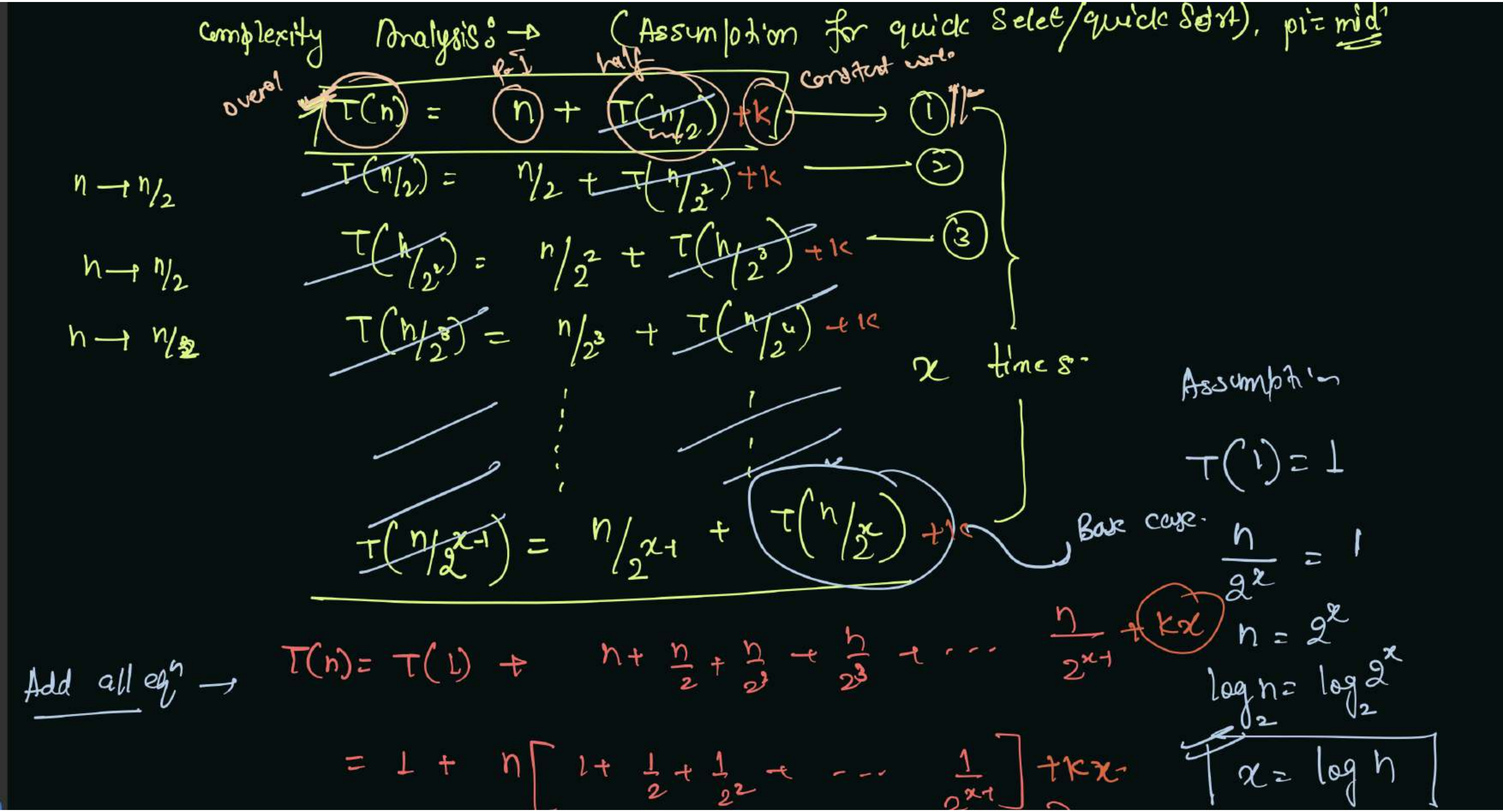
$\boxed{pi == k}$

left side



complexity → $\boxed{O(n)}$

Ratt



Complexity Analysis :→ (Assumption for quick Select/quick Sort), pi = mid

overal, pi, half, constant work

$$T(n) = n + \cancel{T(n/2)} + k \longrightarrow (1)$$

$n \rightarrow n/2$ $$\cancel{T(n/2)} = n/2 + \cancel{T(n/2^2)} + k \longrightarrow (2)$$

$n \rightarrow n/2$ $$\cancel{T(n/2^2)} = n/2^2 + \cancel{T(n/2^3)} + k \longleftarrow (3)$$

$n \rightarrow n/2$ $$\cancel{T(n/2^3)} = n/2^3 + \cancel{T(n/2^4)} + k$$

x times.

$$\vdots$$

$$\cancel{T(n/2^{x-1})} = n/2^{x-1} + T(n/2^x) + k$$

Base case.

Assumption

$$T(1) = 1$$

$$\frac{n}{2^x} = 1$$

$$n = 2^x$$

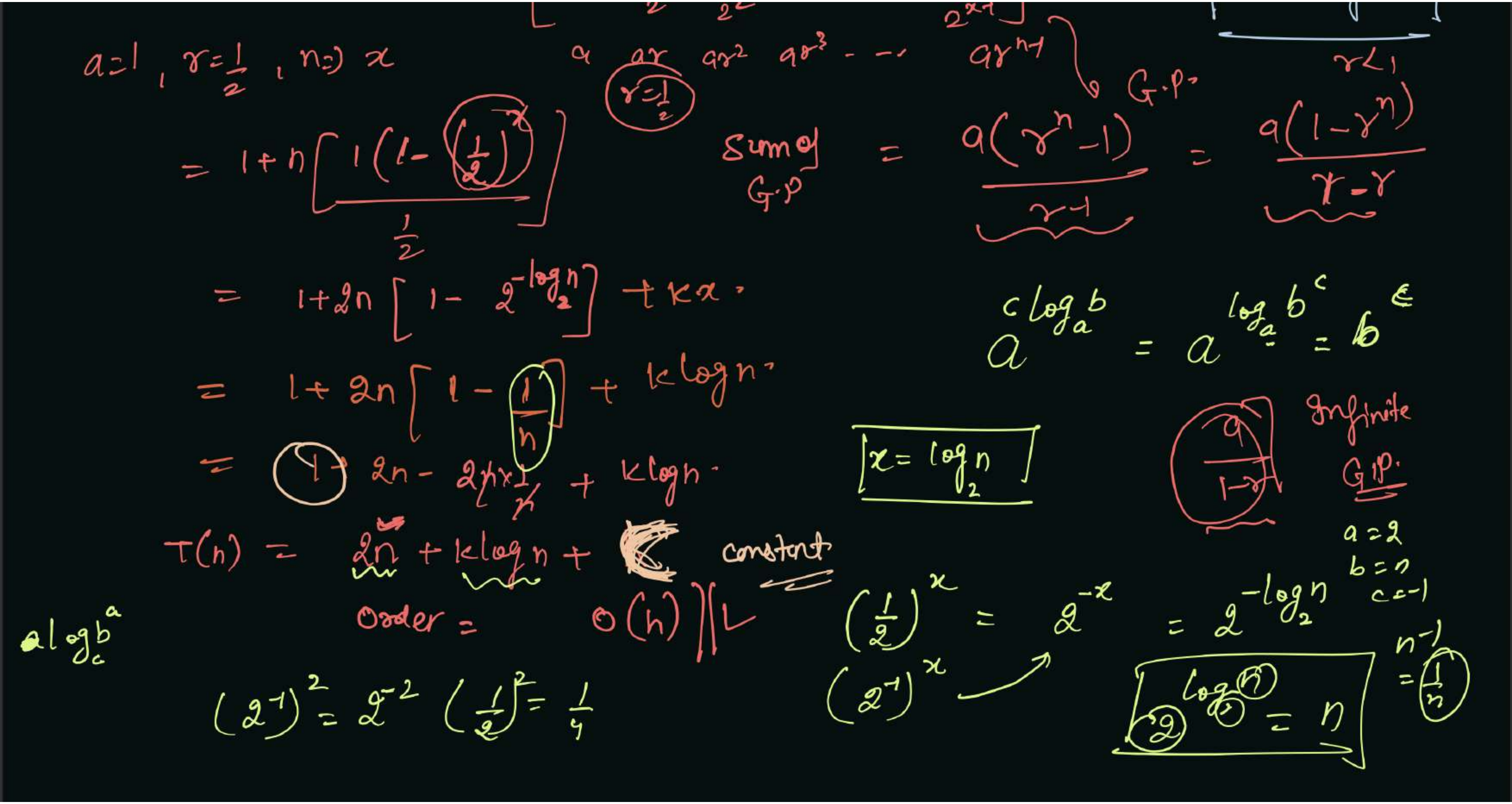
$$\log_2 n = \log_2 2^x$$

$$\boxed{x = \log n}$$

Add all eqn →

$$T(n) = T(1) + n + \frac{n}{2} + \frac{n}{2^3} + \frac{n}{2^3} + \cdots \frac{n}{2^{x-1}} + kx$$

$$= 1 + n\left[1 + \frac{1}{2} + \frac{1}{2^2} + \cdots \frac{1}{2^{x-1}}\right] + kx$$



$\left[\quad 2 \quad 2^2 \quad \cdots \quad 2^{x-1} \right]$

$a=1, \; r=\frac{1}{2}, \; n \Rightarrow x$

$a \quad ar \quad ar^2 \quad ar^3 \; - - - \; ar^{n-1} \rightarrow$ G.P.

$\boxed{r=\frac{1}{2}}$

$r<1$

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

Sum of G.P $= \frac{a(r^n-1)}{\underbrace{r-1}} = \frac{a(1-r^n)}{\underbrace{1-r}}$

$= 1+2n\left[1-2^{-\log_2 n}\right] + kx.$

$a^{c\log_a b} = a^{\log_a b^c} = b^c$

$= 1+2n\left[1-\frac{1}{n}\right] + k\log n.$

Infinite G.P. $\frac{a}{1-r}$

$\boxed{x=\log_2 n}$

$= 1 + 2n - 2n \times \frac{1}{n} + k\log n.$

$T(n) = \underbrace{2n} + \underbrace{k\log n} + C$ constant

$a=2$

$b=n$

$c=-1$

Order $= \Theta(n) \;||\; L$

$a\log_c b^a$

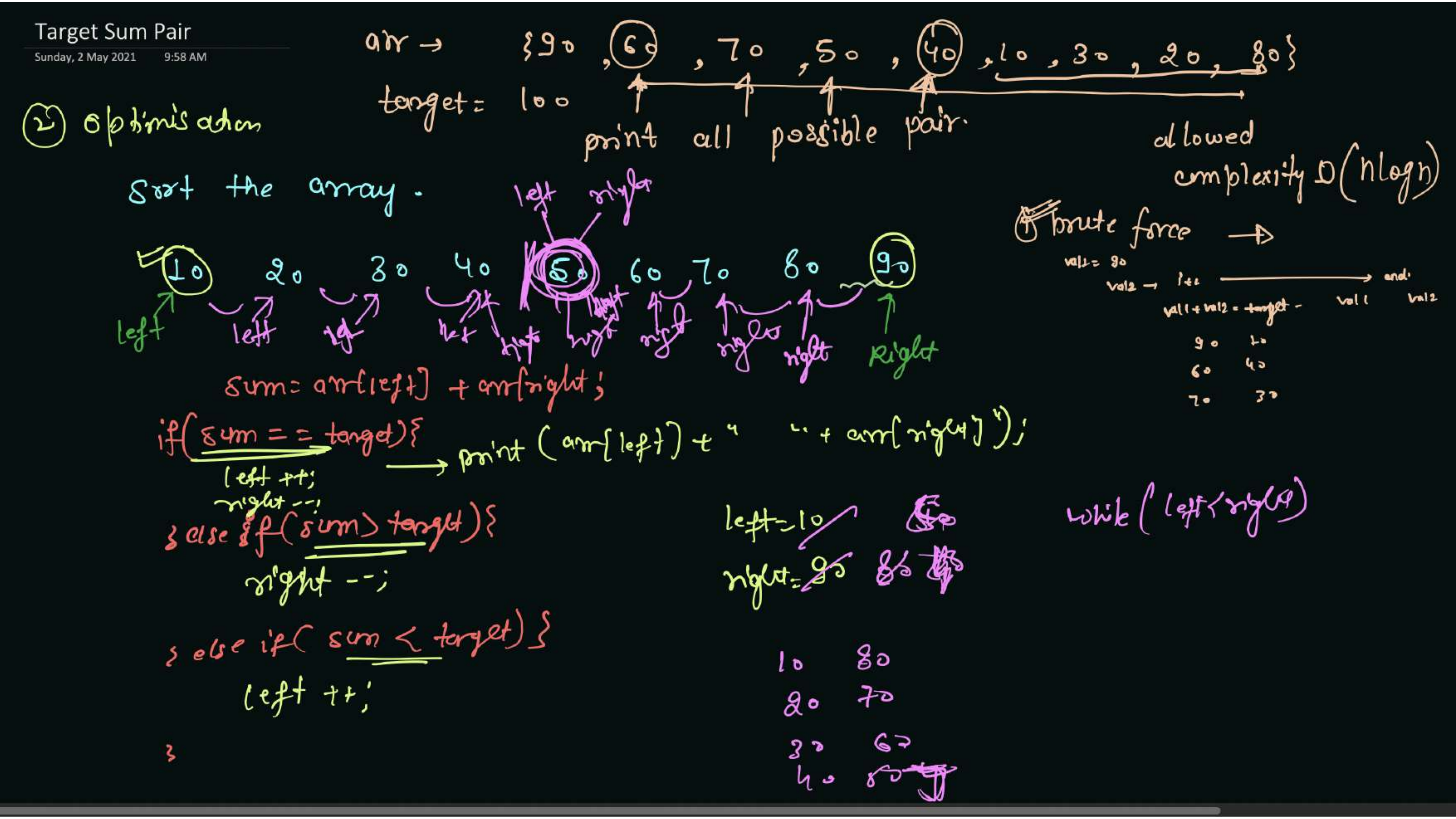
$\left(\frac{1}{2}\right)^x = 2^{-x} = 2^{-\log_2 n}$

$(2^{-1})^x \rightarrow 2^{-x}$

$\boxed{2^{\log_2 n} = n}$

$n^{-1} = \frac{1}{n}$

$(2^{-1})^2 = 2^{-2} \quad \left(\frac{1}{2}\right)^2 = \frac{1}{4}$



Target Sum Pair

Sunday, 2 May 2021 9:58 AM

arr → {90, 60, 70, 50, 40, 10, 30, 20, 80}

target= 100

print all possible pair.

allowed complexity O(nlogn)

① brute force →

val1= 90

val2 → i+1 → end.

val1 + val2 = target -

val1	val2
90	10
60	40
70	30

② optimisation

Sort the array.

10 20 30 40 50 60 70 80 90

left left left left right right right right right Right

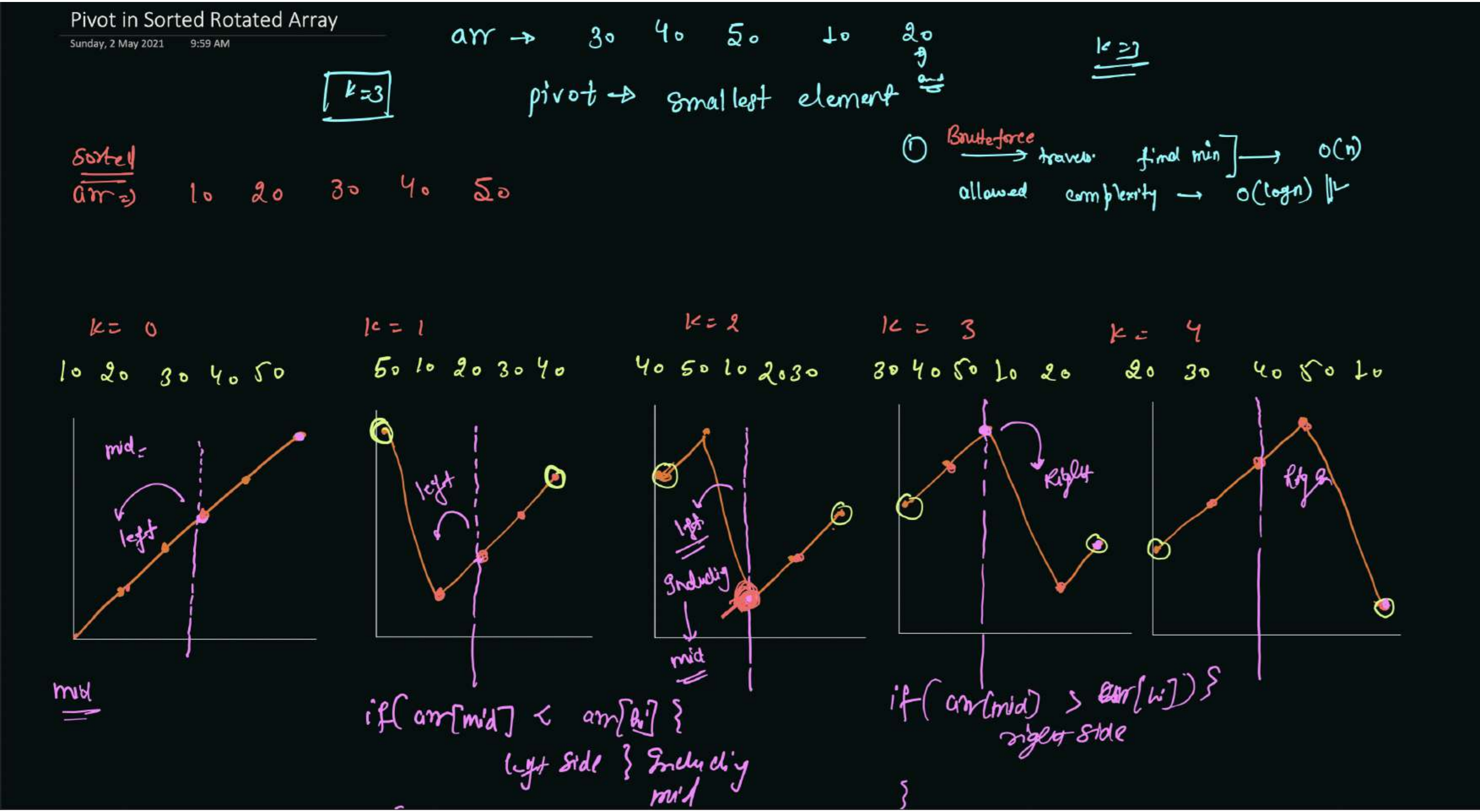
```
sum= arr[left] + arr[right];
if(sum == target){ → print (arr[left] + "  " + arr[right]);
  left ++;
  right --;
}else if(sum> target){
  right --;
}else if(sum < target){
  left ++;
}
```

left= ~~10~~ ~~50~~

right= ~~90~~ 80

while(left<right)

10	80
20	70
30	60
40	60



Pivot in Sorted Rotated Array

Sunday, 2 May 2021 9:59 AM

arr → 30 40 50 10 20

$\boxed{k=3}$

pivot → smallest element

k=3

Sorted arr ⇒ 10 20 30 40 50

① Bruteforce → traverse find min] → O(n)

allowed complexity → O(logn) ||

k = 0

10 20 30 40 50

k = 1

50 10 20 30 40

k = 2

40 50 10 20 30

k = 3

30 40 50 10 20

k = 4

20 30 40 50 10

mid

```
if( arr[mid] < arr[hi] ) {
        left side } including
                    mid
```

```
if( arr(mid) > arr[hi] ) {
        right side
}
```

{ 40 50 60 70 80 90 10 20 30 }

0 1 2 3 4 5 6 7 8

5 6 } → (lo == hi)

↓

ans = arr[lo],

ans = arr[hi].

Right ↑

mid= 5

5, 6

left ↑

mid = 6

5, 8

Right ↑

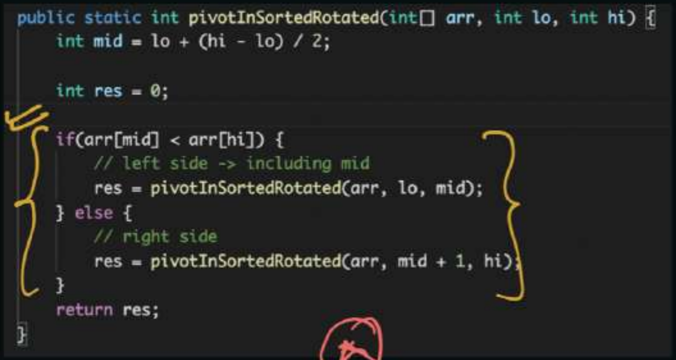
mid=4

0 , 8

answer

distinct → Definetly

ans//



```
public static int pivotInSortedRotated(int[] arr, int lo, int hi) {
    int mid = lo + (hi - lo) / 2;

    int res = 0;

    if(arr[mid] < arr[hi]) {
        // left side -> including mid
        res = pivotInSortedRotated(arr, lo, mid);
    } else {
        // right side
        res = pivotInSortedRotated(arr, mid + 1, hi);
    }
    return res;
}
```

Binary Search

Bubble Sort

Sunday, 2 May 2021 9:58 AM

arr → 40 20 50 30 10

final array = 10 20 30 40 50 after sorting.

① arr → 40 20 50 30 10

1.1 $\overset{20}{\cancel{40}}$ $\overset{40}{\cancel{20}}$ 50 30 10 ↑i

1.2 → 20 40 50 30 10 ↑i

1.3 → 20 40 $\overset{30}{\cancel{50}}$ $\overset{50}{\cancel{30}}$ 10 ↑i

1.4 → 20 40 30 $\overset{10}{\cancel{50}}$ $\overset{50}{\cancel{10}}$ ↑i

n-1 times

20 40 30 10 $\boxed{50}$ Sorted

④

j+i<n

j<n-1

② 2.1 20 40 30 10 $\boxed{50}$ ↑i

2.2 20 $\overset{30}{\cancel{40}}$ $\overset{40}{\cancel{30}}$ 10 $\boxed{50}$ ↑i

2.3 20 30 $\overset{10}{\cancel{40}}$ $\overset{40}{\cancel{10}}$ $\boxed{50}$ i

20 30 10 $\boxed{40 \quad 50}$ sorted ③

③ 3.1 20 30 10 $\boxed{40 \quad 50}$ ↑i

3.2. 20 $\overset{10}{\cancel{30}}$ $\overset{30}{\cancel{10}}$ $\boxed{40 \quad 50}$ ↑index

20 10 $\boxed{30 \quad 40 \quad 50}$ Sorted ②

④ 4.1 $\overset{10}{\cancel{20}}$ $\overset{20}{\cancel{10}}$ $\boxed{30 \quad 40 \quad 50}$ ③ ↑index

10 $\boxed{20 \quad 30 \quad 40 \quad 50}$ sorted ①

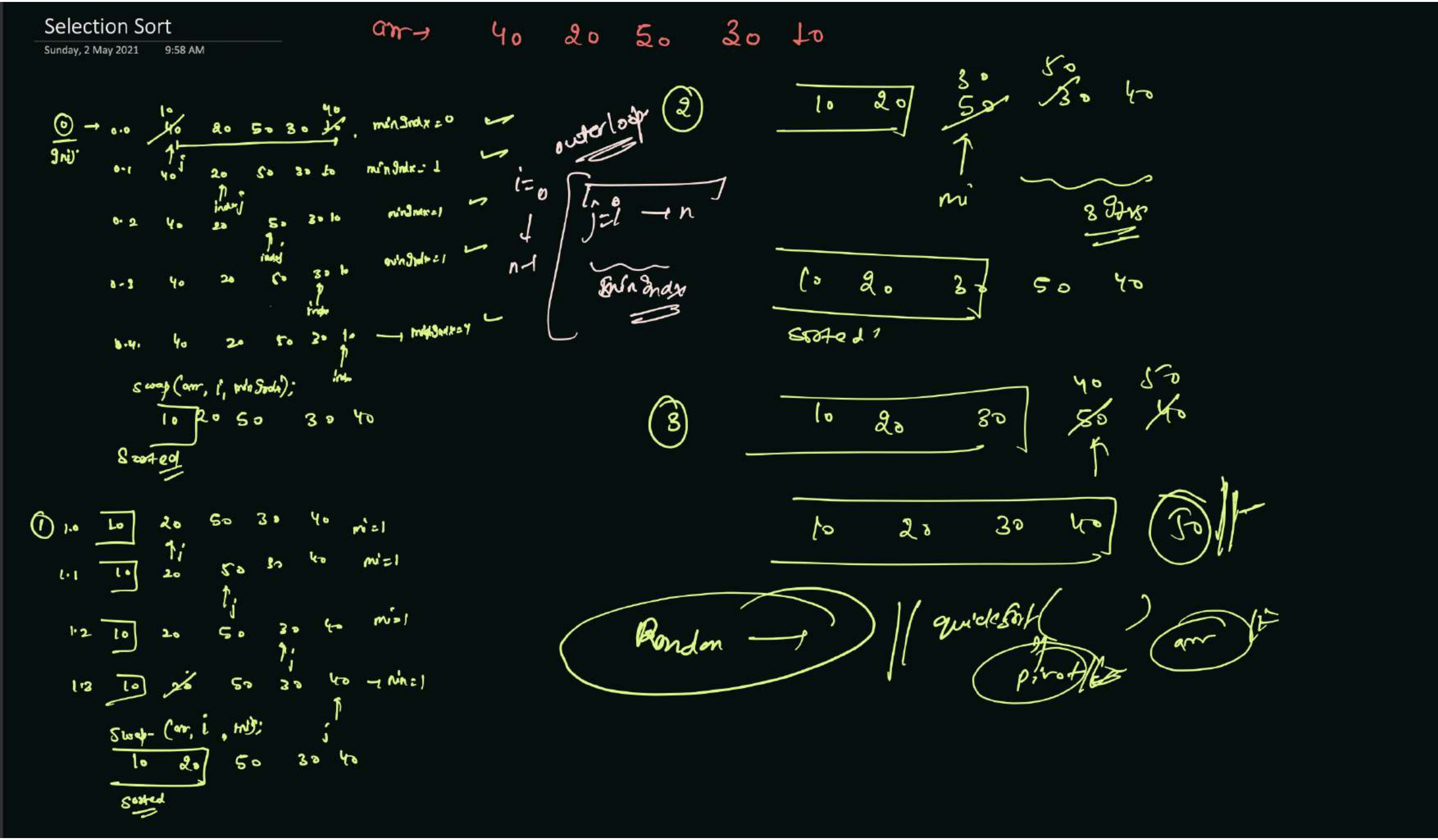
< n-1

0, 1, 2, 3 < 4

outer loop: inner loop; j=0; arr.length-1-i; j++

i=0

i<arr.length



Selection Sort

Sunday, 2 May 2021 9:58 AM

arr → 40 20 50 30 10

0 → 0.0 ~~40~~ 10 20 50 30 ~~10~~ 40 . minIndx = 0

Imij

0.1 40 20 50 30 10 minIndx = 1

0.2 40 20 50 30 10 minIndx=1

0.3 40 20 50 30 10 minIndx=1

0.4 40 20 50 30 10 → minIndx=4

swap(arr, i, minIndx);

10 20 50 30 40

Sorted

1 1.0 10 20 50 30 40 mi=1

1.1 10 20 50 30 40 mi=1

1.2 10 20 50 30 40 mi=1

1.3 10 ~~20~~ 50 30 40 → min=1

Swap (arr, i, mi);

10 20 50 30 40

sorted

outerloop 2

i=0 ↓ n-1

i
j=i → n
minIndex

10 20 | 30 ~~50~~ 50 ~~30~~ 40

mi

10 20 30 50 40

sorted

3 10 20 30 | 40 ~~50~~ 50 ~~40~~

10 20 30 40 | 50

Random → || quicksort(pivot) arr