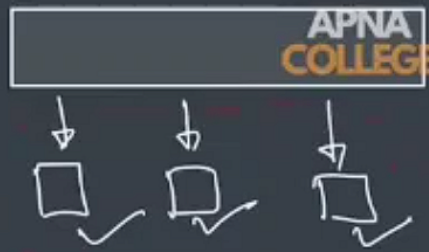
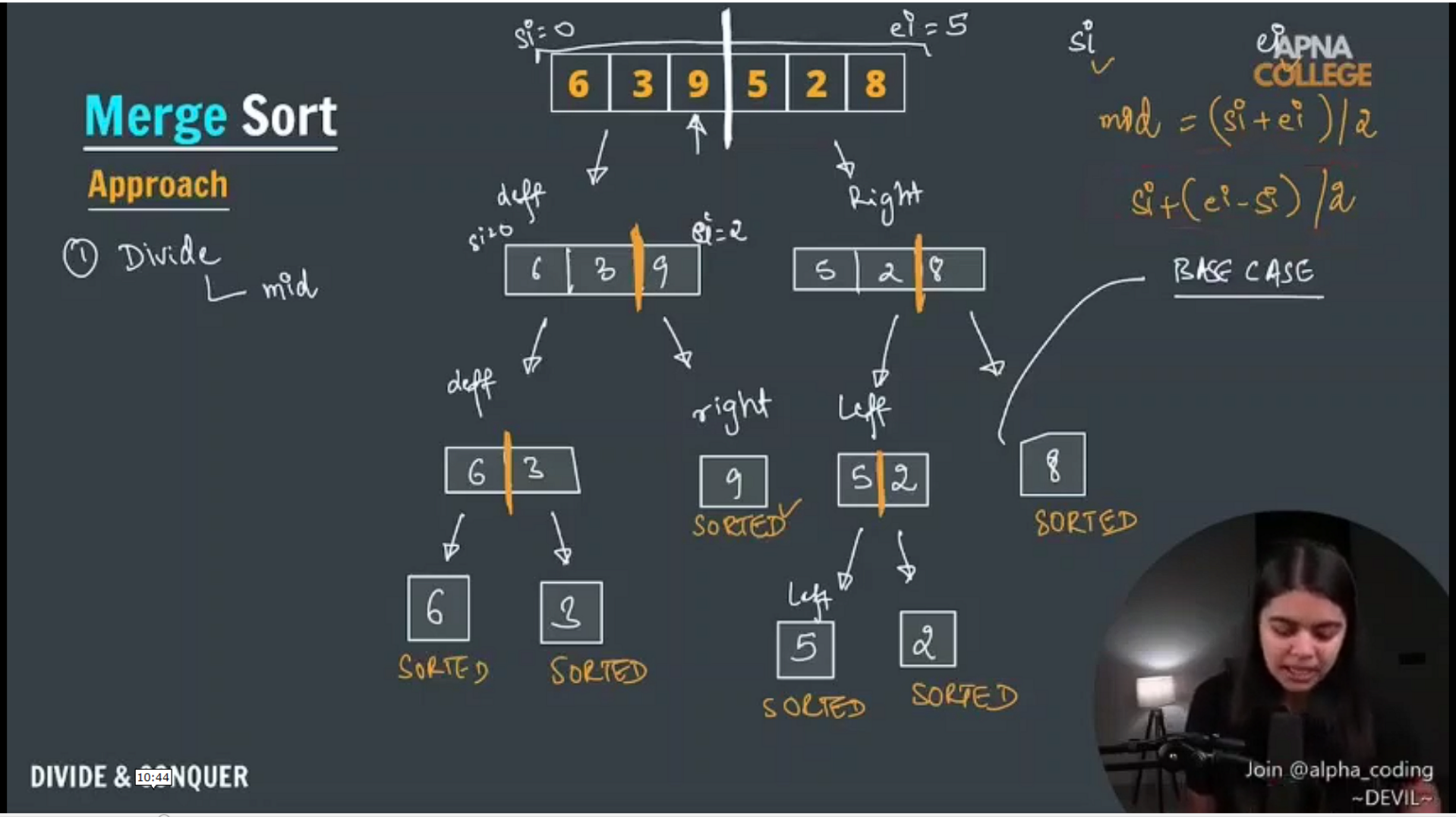
Divide & Conquer

Algorithms

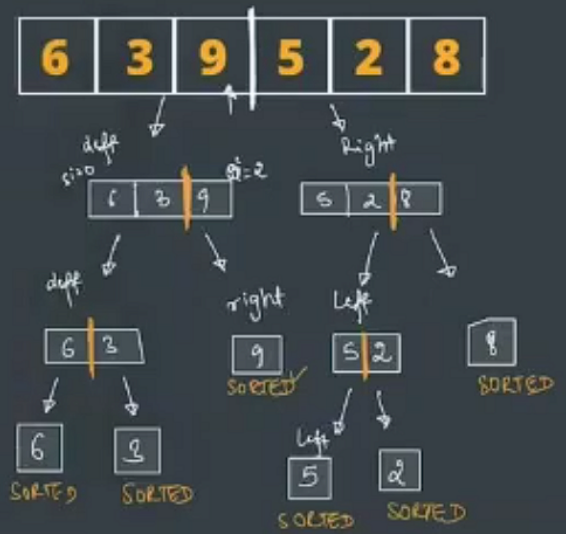


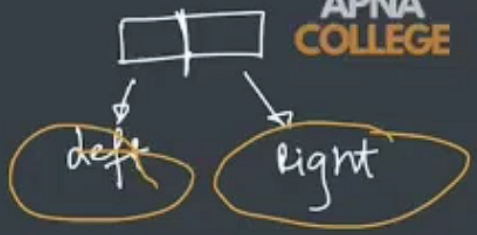


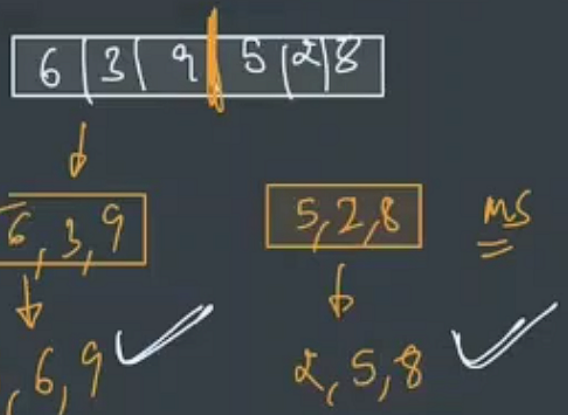
Merge Sort

Approach

1. Divide
   - mid
2. mergeSort(left)
   mergeSort(right)
3. merge







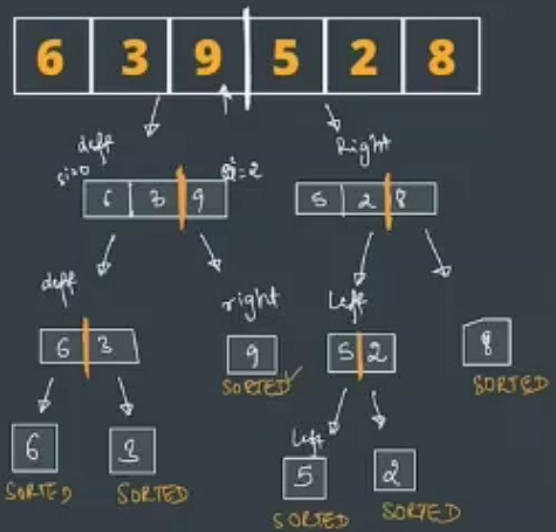
DIVIDE & CONQUER

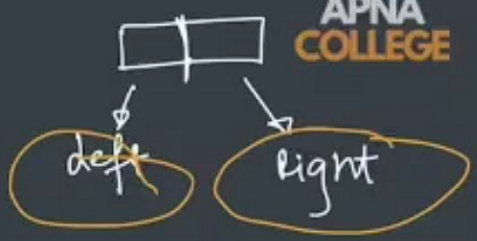
Merge Sort

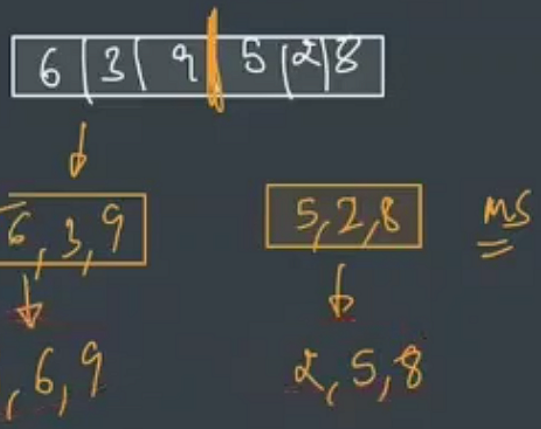
Approach

(1) Divide
└ mid

(2) mergeSort(left)
mergeSort(right)







DIVIDE & CONQUER

Merge Sort

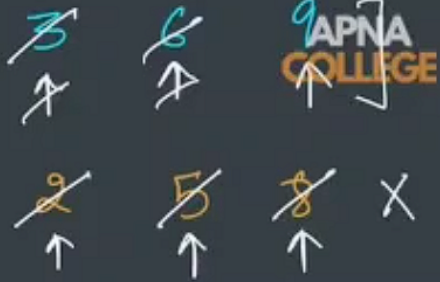
Approach

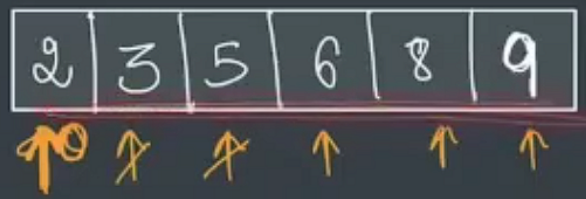
(1) Divide
└ mid

(2) mergeSort(left)
mergeSort(right)

(3) merge

(3,6,9) (2,5,8) MERGE



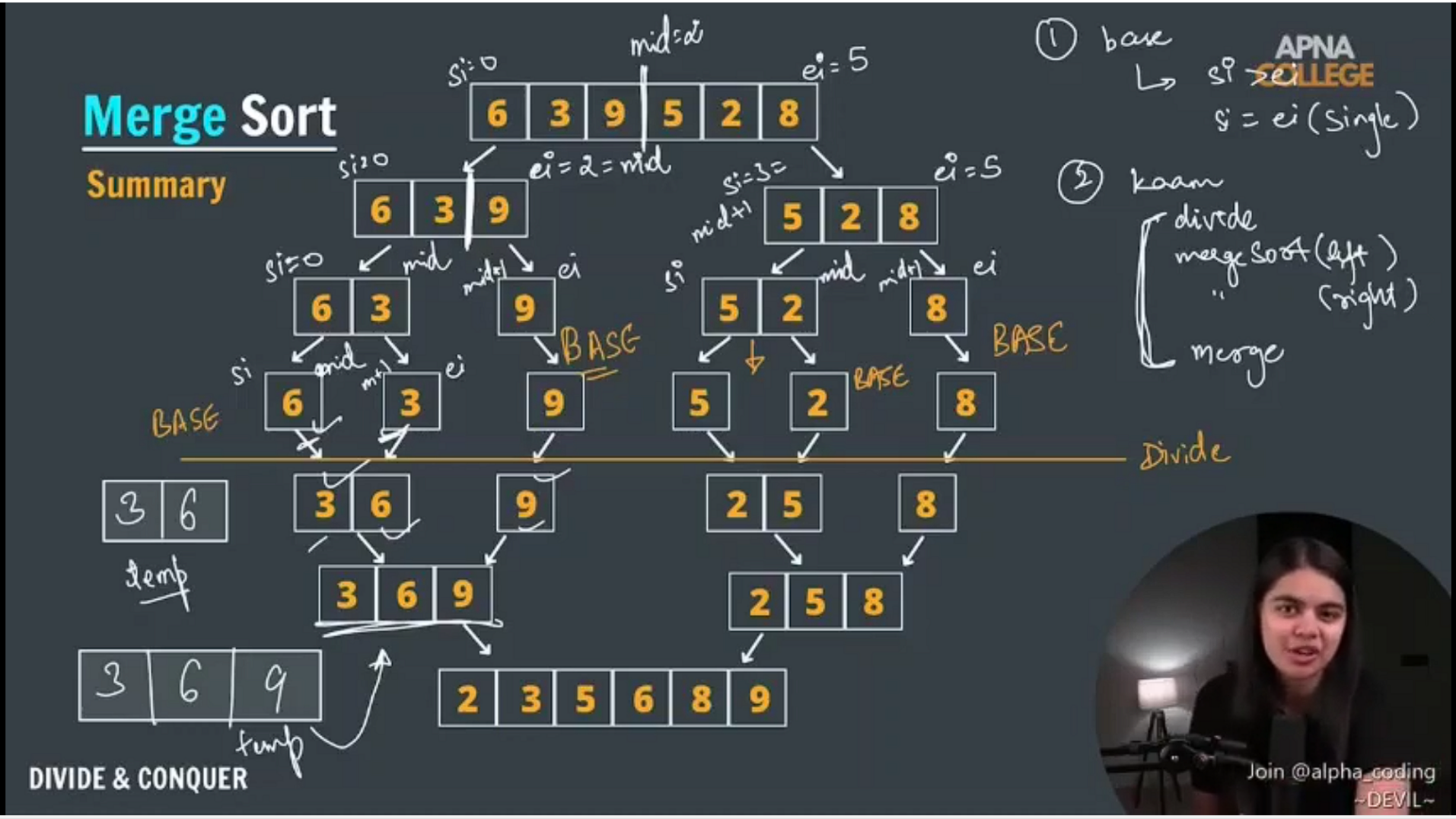


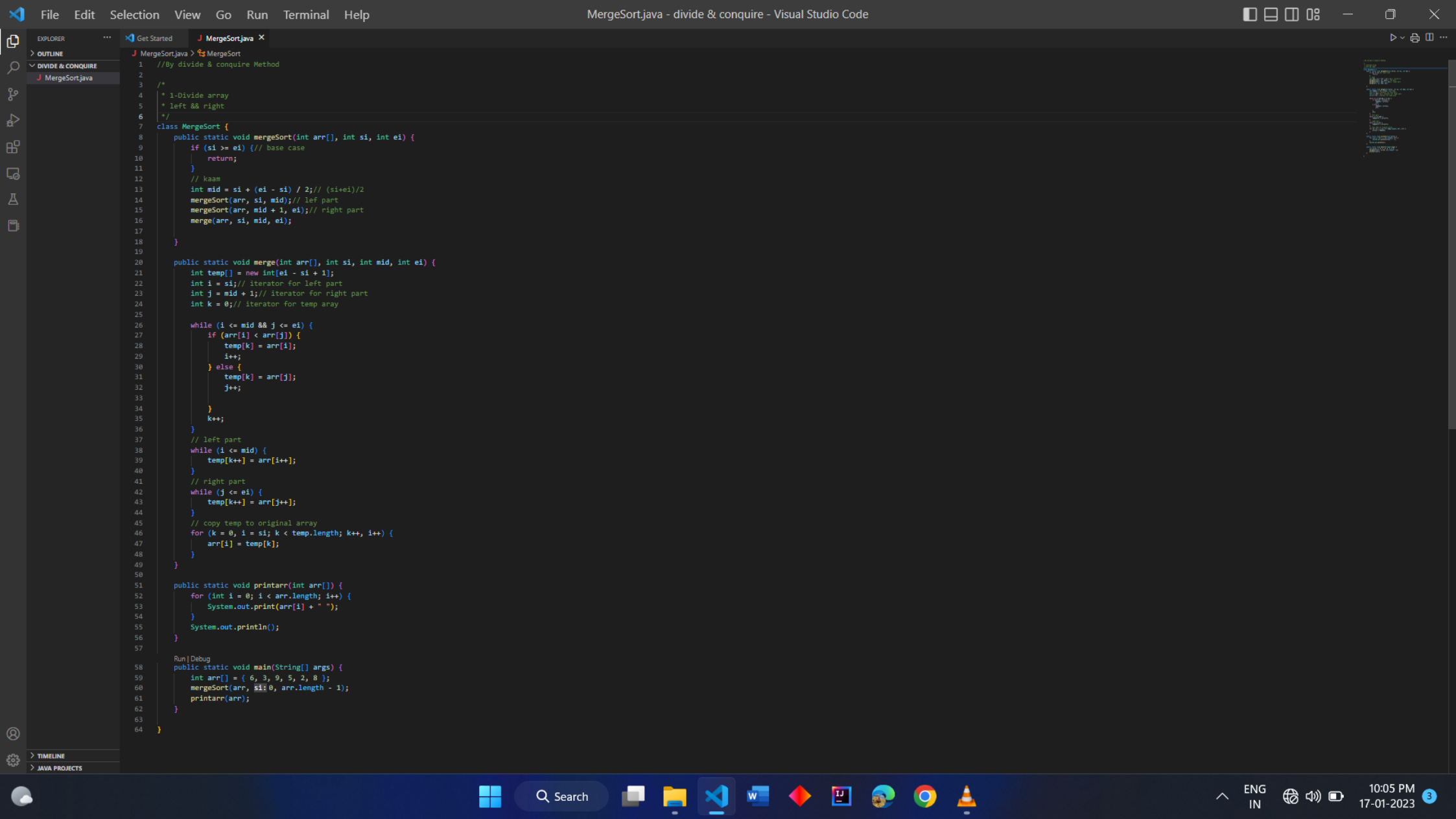
temp → Left els + Right els.
size

DIVIDE & CONQUER

Merge Sort

Summary




File Edit Selection View Go Run Terminal Help
MergeSort.java - divide & conquire - Visual Studio Code
EXPLORER
OUTLINE
DIVIDE & CONQUIRE
MergeSort.java
Get Started
MergeSort.java
MergeSort.java > MergeSort
//By divide & conquire Method
/*
* 1-Divide array
* left && right
*/
class MergeSort {
public static void mergeSort(int arr[], int si, int ei) {
if (si >= ei) {// base case
return;
}
// kaam
int mid = si + (ei - si) / 2;// (si+ei)/2
mergeSort(arr, si, mid);// lef part
mergeSort(arr, mid + 1, ei);// right part
merge(arr, si, mid, ei);
}
public static void merge(int arr[], int si, int mid, int ei) {
int temp[] = new int[ei - si + 1];
int i = si;// iterator for left part
int j = mid + 1;// iterator for right part
int k = 0;// iterator for temp aray
while (i <= mid && j <= ei) {
if (arr[i] < arr[j]) {
temp[k] = arr[i];
i++;
} else {
temp[k] = arr[j];
j++;
}
k++;
}
// left part
while (i <= mid) {
temp[k++] = arr[i++];
}
// right part
while (j <= ei) {
temp[k++] = arr[j++];
}
// copy temp to original array
for (k = 0, i = si; k < temp.length; k++, i++) {
arr[i] = temp[k];
}
}
public static void printarr(int arr[]) {
for (int i = 0; i < arr.length; i++) {
System.out.print(arr[i] + " ");
}
System.out.println();
}
Run | Debug
public static void main(String[] args) {
int arr[] = { 6, 3, 9, 5, 2, 8 };
mergeSort(arr, si: 0, arr.length - 1);
printarr(arr);
}
}
TIMELINE
JAVA PROJECTS
Search
ENG
IN
10:05 PM
17-01-2023

Dry Run

```java
public static void mergeSort(int arr[], int si, int ei) {
    if(si >= ei) {
        return;
    }
    int mid = si + (ei - si)/2; // or = (si + ei) / 2;
    mergeSort(arr, si, mid);
    mergeSort(arr, mid+1, ei);

    merge(arr, si, mid, ei);
}
```

Time O(nlogn) ✓

Space O(n) ✓

Depth first MS

```java
//merge method to merge the sorted parts
public static void merge(int arr[], int si, int mid, int ei) {
    int temp[] = new int[ei-si+1];
    int i = si; //idx for 1st sorted part
    int j = mid+1; //idx for 2nd sorted part
    int k = 0; //idx for temp;

    while(i <= mid && j <= ei) {
        if(arr[i] < arr[j]) {
            temp[k] = arr[i];
            i++;
        } else {
            temp[k] = arr[j];
            j++;
        }
        k++;
    }

    //for leftover elements of 1st sorted part
    while(i <= mid) {
        temp[k++] = arr[i++];
    }

    //for leftover elements of 2nd sorted part
    while(j <= ei) {
        temp[k++] = arr[j++];
    }

    //copy temp to original array
    for(k=0, i=si; k<temp.length; k++, i++) {
        arr[i] = temp[k];
    }
}
```

Merge Sort

Quick Sort

- average $O(n \log n)$
- worst $O(n^2)$
- Space $O(1)$

DIVIDE & CONQUER

Quick Sort

- average $O(n \log n)$
- worst $O(n^2)$
- Space $O(1)$

DIVIDE & CONQUER

6	3	9	8	2	5

APNA COLLEGE

Pivot & Partition

1. Pivot → random / median / first / last

DIVIDE & CONQUER

APNA COLLEGE

Pivot & Partition

1. Pivot (last element)
2. Partition (parts)
3. Quick Sort (left)
   " (right)

3, 2 < 5 < 6, 9, 8

quick Sort — quick Sort

Base

Single → ✓ SORTED

DIVIDE & CONQUER

si=0 ei=5

6	3	9	8	2	5

j=0 j=1 j=2 j=3 j=4

pivot = 5

ALPHA COLLEGE

Pivot & Partition

1. Pivot (last element)
2. Partition (parts)
3. Quick sort (left)
   " (right)

```
i++;
//swap
int temp= arr[j]
arr[j]= arr[i]
arr[i]= temp
```

i = -1 0 1

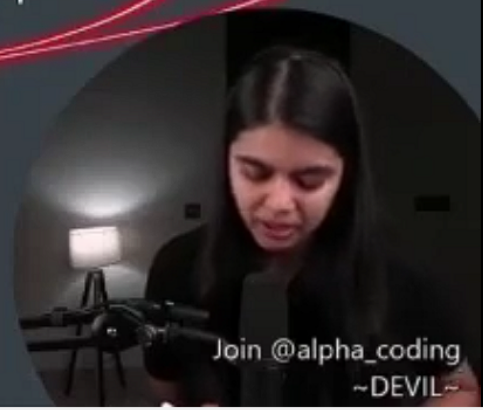
3	2			6	

-1 0 1 2 3 4 5

(6) (3) (9) (8) (5)

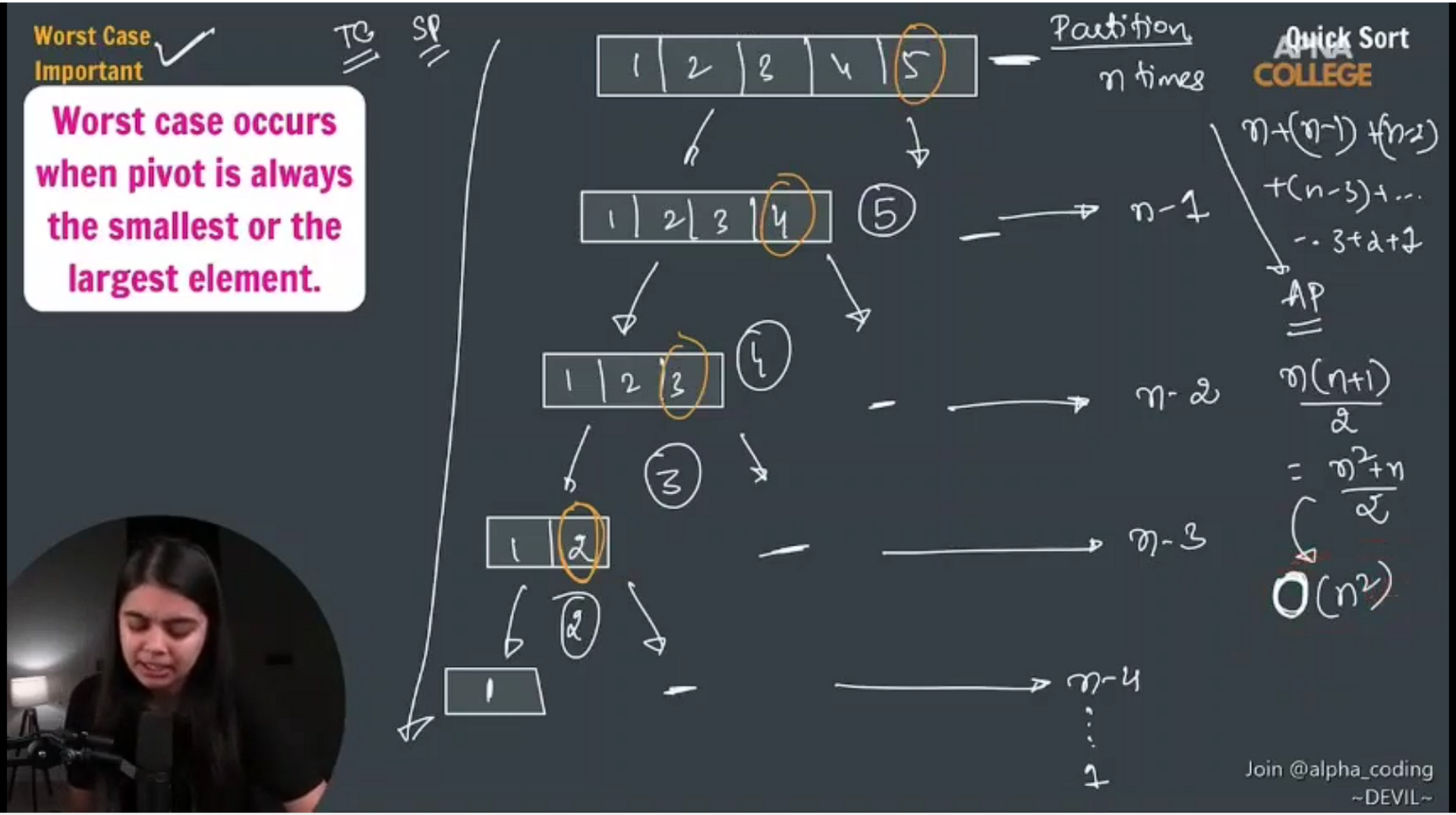
3	2	9	8	6	5

DIVIDE & CONQUER



Lecture 16(Divide & Conquer)_ Alpha batch_ Java full course _ Java +DSA+ Interview_ Placement course.mp4 - VLC media player

Media Playback Audio Video Subtitle Tools View Help



55:42 1:17:58

92%

Search

ENG IN

11:12 PM 21-01-2023

APNA COLLEGE

Search in Rotated Sorted Array

input : sorted, rotated array with distinct numbers (in ascending order)
It is rotated at a pivot point. Find the index of given element.

4	5	6	7	0	1	2

target : 0

output : 4

DIVIDE & CONQUER

APNA COLLEGE

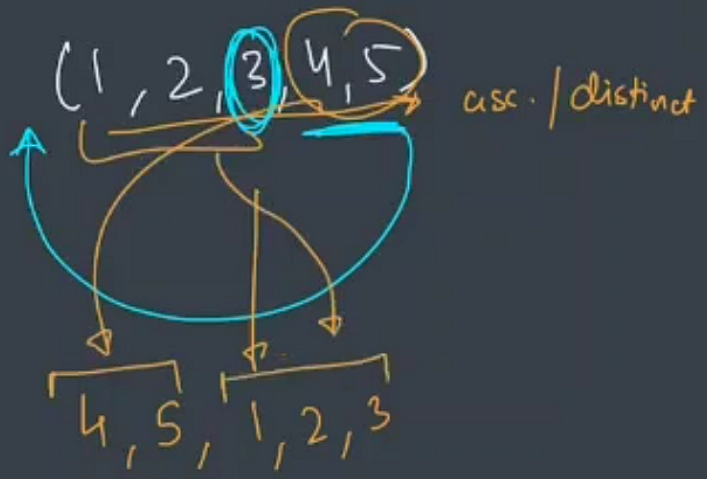
Search in Rotated Sorted Array

input : sorted, rotated array with distinct numbers (in ascending order)
It is rotated at a pivot point. Find the index of given element.

4	5	6	7	0	1	2

target : 0

output : 4

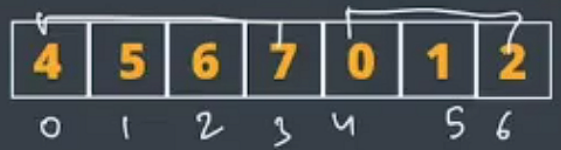


DIVIDE & CONQUER

APNA COLLEGE

Search in Rotated Sorted Array

input : sorted, rotated array with distinct numbers (in ascending order)
It is rotated at a pivot point. Find the index of given element.



target : 0

output : 4

0, 1, 2, 4, 5, 6, 7

DIVIDE & CONQUER

Search in Rotated Sorted Array

→ Divide & Conquer

APNA COLLEGE

input : sorted, rotated array with distinct numbers (in ascending order)
It is rotated at a pivot point. Find the index of given element.

4	5	6	7	0	1	2

target : 0

linear search O(n)

output : 4

nlogn

DIVIDE & CONQUER

APNA COLLEGE

Search in Rotated Sorted Array

input : sorted, rotated array with distinct numbers (in ascending order)
It is rotated at a pivot point. Find the index of given element.

4	5	6	7	0	1	2

target : 0

output : 4

DIVIDE & CONQUER

Bn

mid = (si+ei)/2

APNA COLLEGE

Search in Rotated Sorted Array

Approach

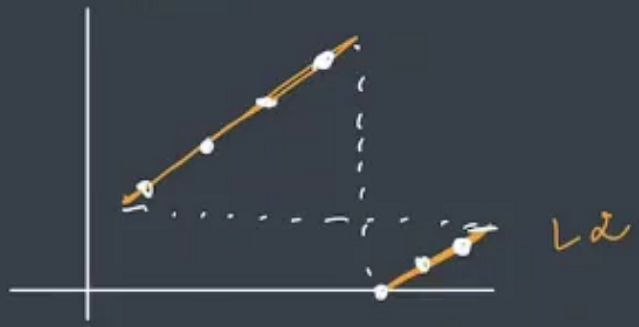
si

4	5	6	7	0	1	2

ei

tar = 0

L1



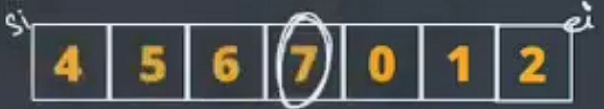
L2

DIVIDE & CONQUER

Search in Rotated Sorted Array

APNA COLLEGE

Approach



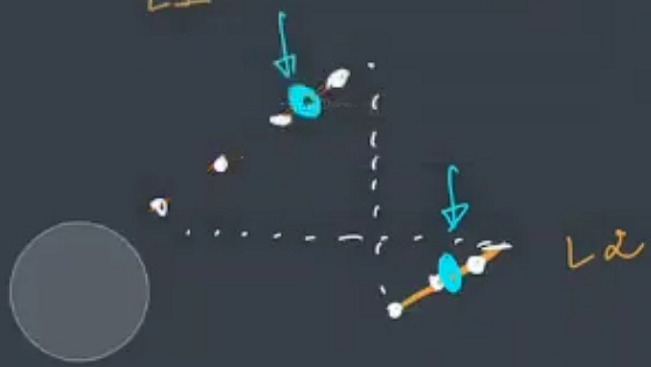
tar = 0

mid = (si+ei)/2
arr[mid]

L1 L2

(si <= tar <= mid) (mid =< tar <= ei)

diff Right (false) right left (false)

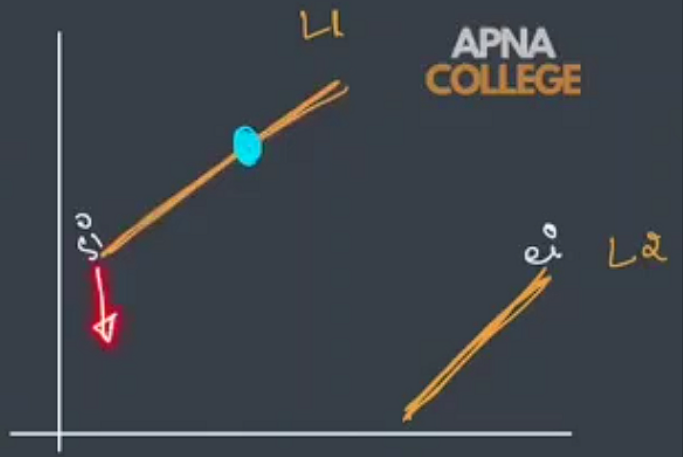


DIVIDE & CONQUER

Search in Rotated Sorted Array

Approach

tar = 0



case 1: mid on L1 arr[si] <= mid

case a: d1 left (si =< tar <= mid)

case b: ~~d1~~ mid right else

case 2: mid on L2

case c: d1 right (mid =< tar <= ei)

case d: mid left else

DIVIDE & CONQUER