Binary Search

It is one of the most popular technique (Algorithm) for Searching

Prerequisites For Binary Search :

* Your Array must be in Sorted Order either Ascending or Descending Order .

It helps to Reduce the Time Complexity

Linear Search Time Complexity 🡺 O(n)

Binary Search Time Complexity 🡺 O(log n)

Suppose we have an Sorted Array (Ascending) :

Arr = [2,4,6,8,12,25,30,37,47)

So our array is lower to upper bound

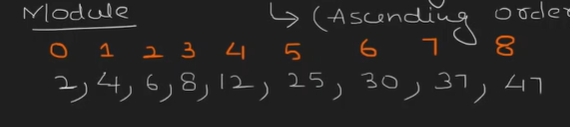
We want to search the 37 element index

Linear Search Takes so many iteration , so we go with Binary Search .

Binary Search works on Divide & Conquer Rule (Module)

Divide & Conquer means Takes a Big Problem Broke into Small problems and solve them .

Our Array is :



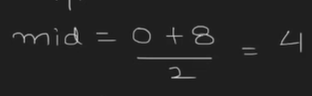
Our Searching item is 🡺 37

Divide & Conquer in Binary Search :

* Array is in the Sorted Way
* Takes the Mid Index (value) :

Formula For Mid 🡺 (Starting Index + Ending Index) / 2

Mid = 0 + 8 / 2

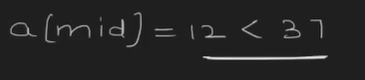


Mid Index 🡺 4

Arr[4] = 12

Now we see Our array is divide into two array using mid value left and right

So 12 < 37



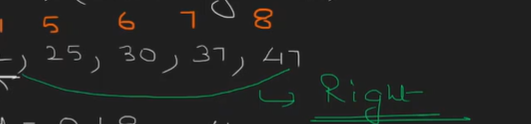
So our search item is greater than mid , so our value is on right side of my right

* Takes the new mid value based on the condition :

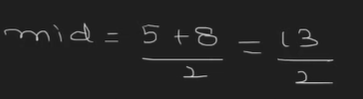
If our mid is less than searching item we skip the left part and take the new Mid Value on Right side

Suppose our Mid is Greater than Searching item we skip the right side and take new mid value on left side

Now based on the condition we take new mid value between 5 to 8



Mid 🡺 5+8/2

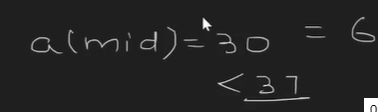


Now our Mid value is 6 .

Arr[6] 🡺 30

So again our mid is less than search item

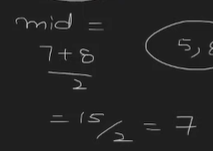
30 < 37



* Again Take New mid value on Right side :

Now we take 7 and 8

Mid 🡺 7 +8 /2

* 15/2
* 7
* 

Mid value = search index

Arr[7]=37 ; Searching item =37

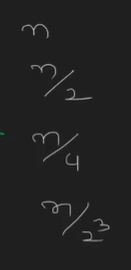
Now our Mid value is the index value of the searching item

Now 7th index is 37 we find with 3 steps but in linear search it takes 7 steps

So Our Time Complexity is :

We divide our array as 2 , 4 , 8 …. Until we find

So it is Log n

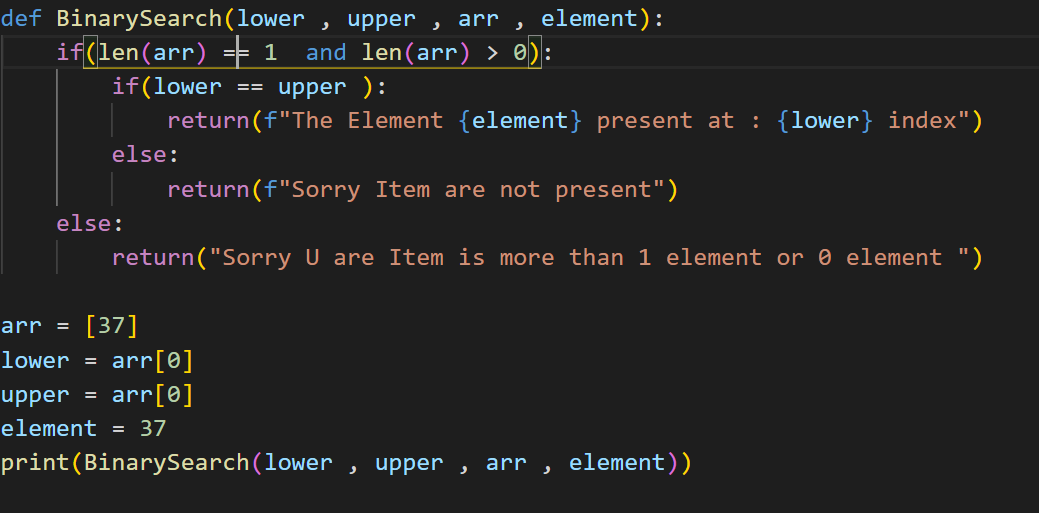


Time Complexity is 🡺 O(log n)

Important Concept :

If we Have only one element in an array then lower and upper bond is same I ==j

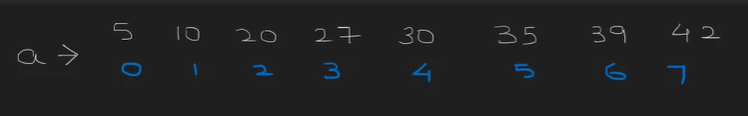
Code For One Element in an Array :



The time Complexity for the above code is constant O(1)

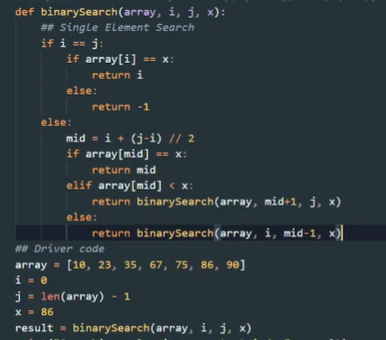
Code for Binary Search:

The Array we have is :



Now we see it is in Sorted;

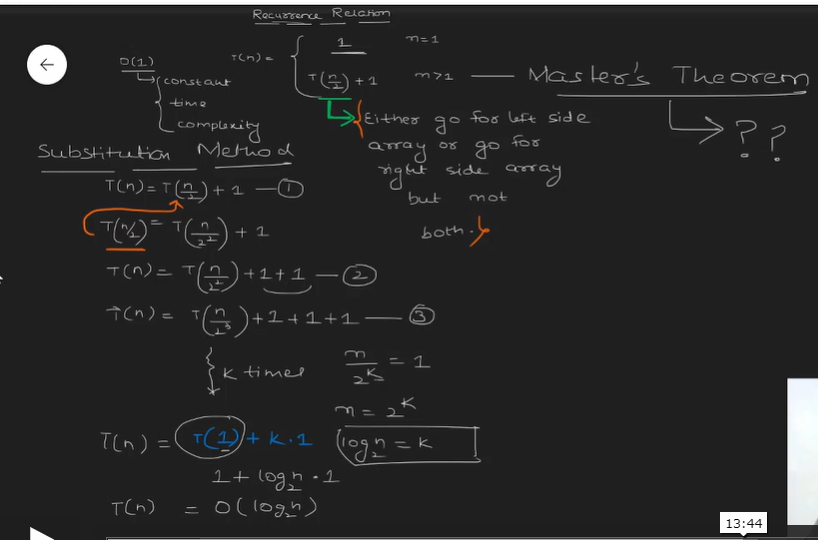
The Lower Bond is 0 and upper bond is 7



The time Complexity for the above code is constant O(log n )

Here we Use Recursive So we need To Check The Time Complexity Of Recursive

Of Above code



Binary Search without Recursive :

