Find Assam? Or way to TN to Assam?

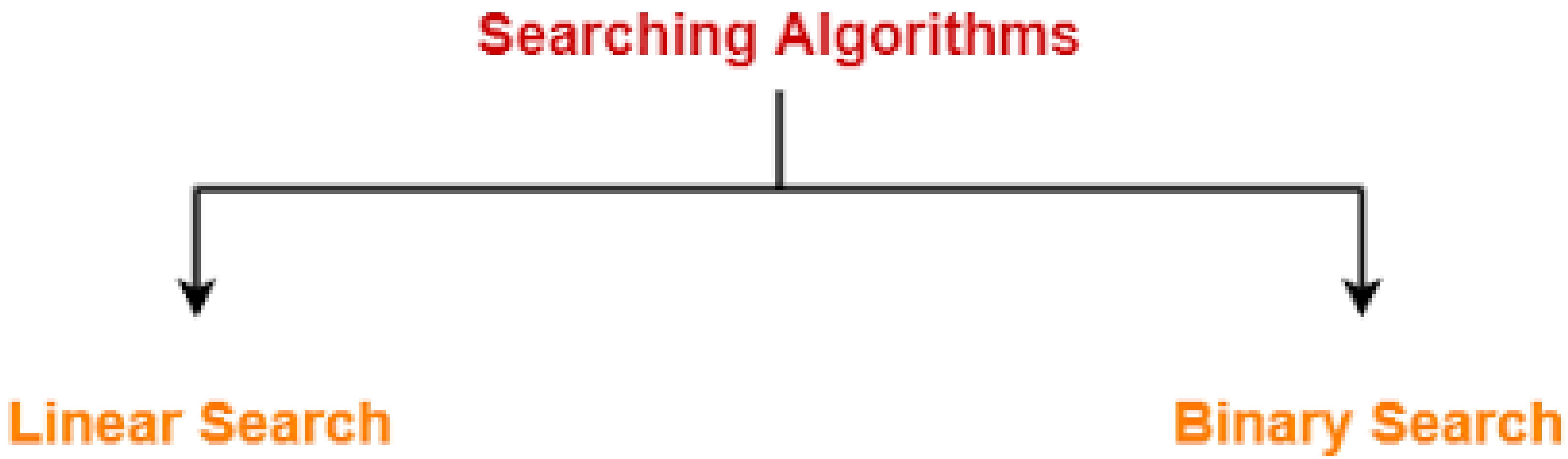
Route finding problem

Tour to Manali

Touring Problems

Searching

The process of finding a given value **position** in a list of values. It decides whether a search key is present in the **data** or not.



Binary search

A **Binary search** algorithm finds the position of a specified input value (the search "key") within a sorted array . For binary search, the array should be arranged in ascending or descending order.

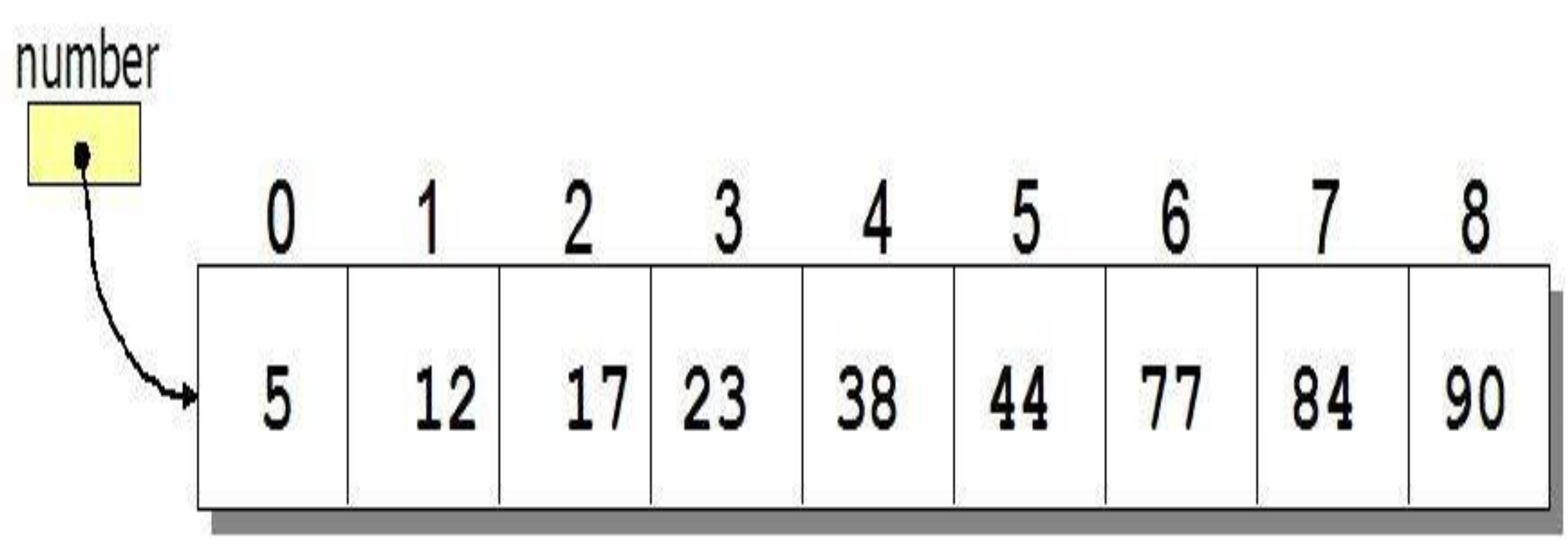
Why Binary Search Is Better Than Linear Search ?

A linear search works by **looking at each element in a list** of data until it either finds the target or reaches the end. This results in **O(n) performance** on a given list.

A binary search comes with the prerequisite that the **data must be sorted**. We can use this information to decrease the number of items we need to look at to find our target.

That is ,Binary Search is **fast as compared to linear search** because of **less number** of **computational operations** .

Example :



Step 1

	low	high	mid
#1	0	8	4

search(44)

$$mid = \left\lfloor \frac{low + high}{2} \right\rfloor$$

0	1	2	3	4	5	6	7	8
5	12	17	23	38	44	77	84	90
low				mid				high

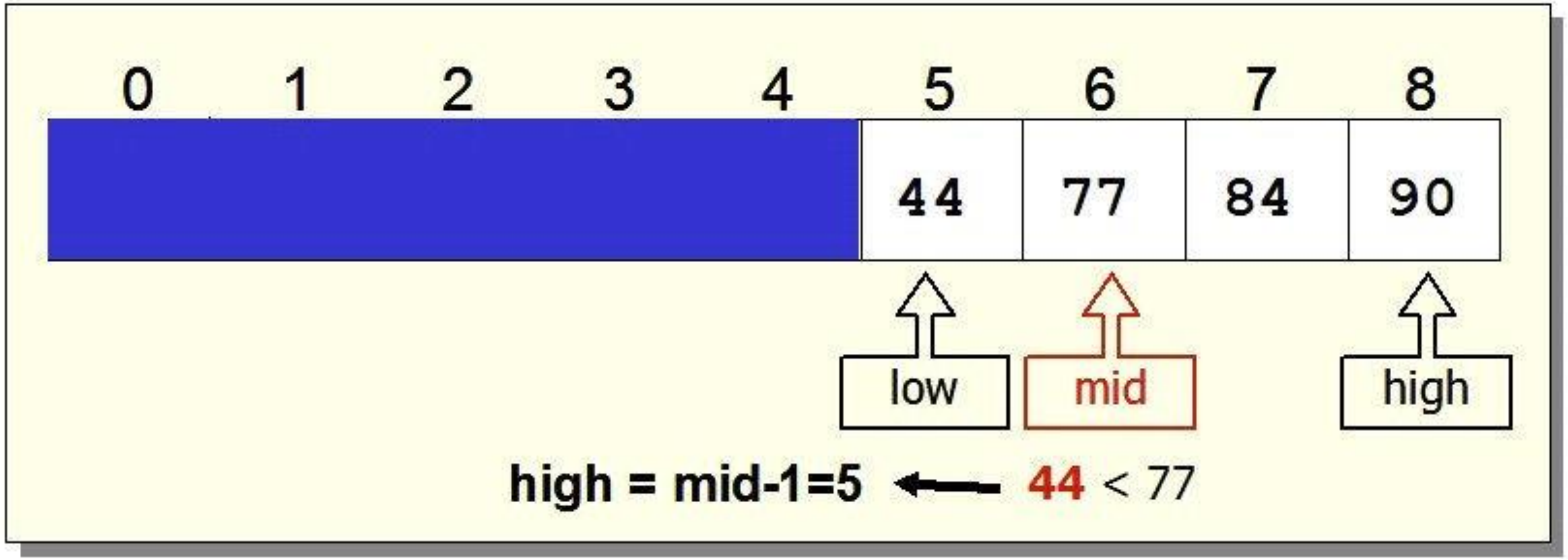
38 < **44** ⟶ **low = mid+1 = 5**

Step 2

	low	high	mid
#1	0	8	4
#2	5	8	6

search(44)

$$mid = \left\lfloor \frac{low + high}{2} \right\rfloor$$

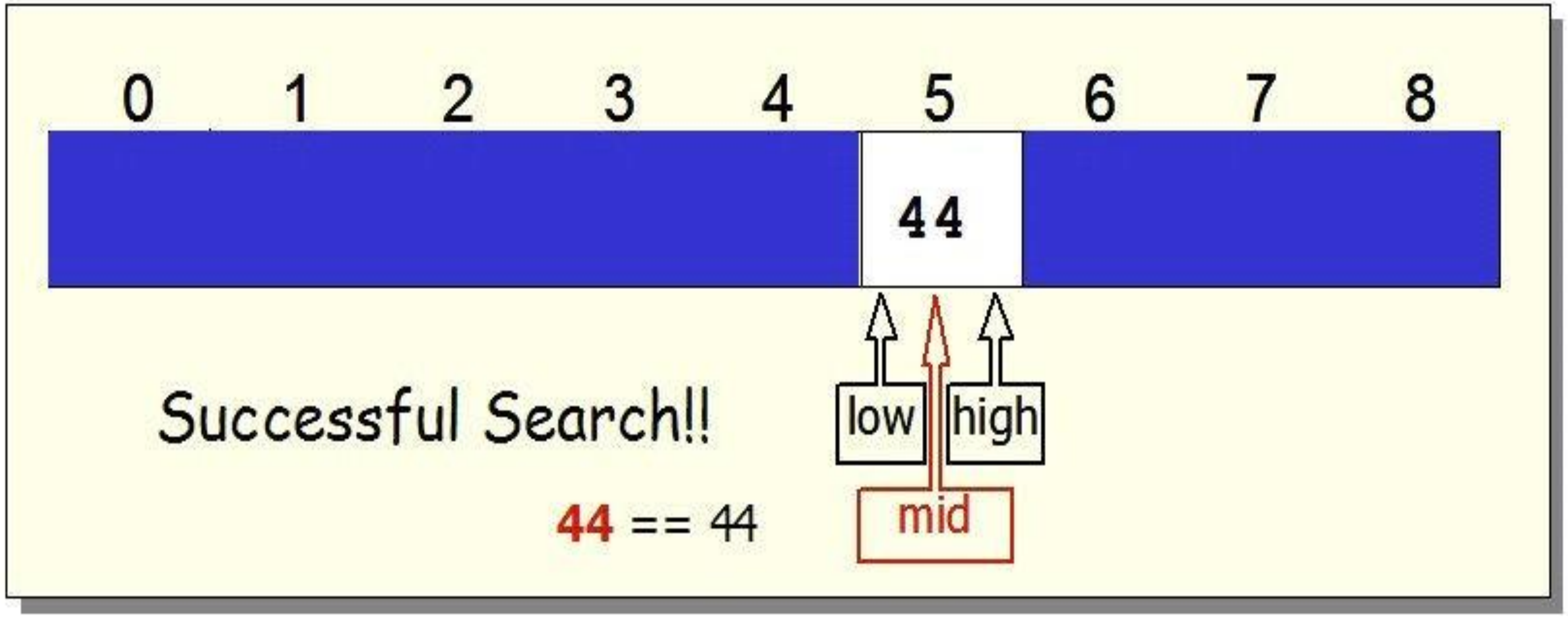


Step 3

	low	high	mid
#1	0	8	4
#2	5	8	6
#3	5	5	5

search(44)

$$mid = \left\lfloor \frac{low + high}{2} \right\rfloor$$



ALGORITHM (Recursive)

recursivebinarysearch(int A[], int first, int last, int key)

if last<first index=-1

else

int **mid = (first + last) / 2**

if key=A[mid]

index=mid

else if **key<A[mid]**

index= **recursivebinarysearch(A, first, mid – 1, key)** else

index= **recursivebinarysearch(A, mid + 1, last, key)**

return index

Analysis

Linear search runs in ***O(n)*** time. Whereas binary search produces the result in ***O(log n)*** time

Let **T(n)** be the number of comparisons in worst-case in an array of **n** elements.

Hence,
$T(n)=\{0T(n2)+1 if n=1 otherwise$
Using this recurrence relation $T(n)=logn$

Therefore, binary search uses $\boldsymbol{O(logn)}$

Assessment

1.Given an array arr = {45,77,89,90,94,99,100} and key = 99; what are the mid values(corresponding array elements) in the first and second levels of recursion?

a) 90 and 99

b) 90 and 94

c) 89 and 99

d) 89 and 94

2. What is the average case time complexity of binary search using recursion?

a) O(nlogn)

b) O(logn)

c) O(n)

d) O(n^2)


Thank You