# *Binary Search*

By : Shivansh ( CF : shiv_codegen )

# Problem :

Given an array of numbers, find the index of some element X in it. (if it exists)

| 56 | 1 | 23 | 45 | 111 | 9 | 31 | 47 |
|----|---|----|----|-----|---|----|----|

What will you do ?

# Any other search algorithm ?

- Given the array is sorted. Can we avoid useless comparisons?

- If arr[i] < target, what does it tell us about arr[index] such that index < i ?

- It turns out that we can eliminate certain indices without actually comparing them with our target.

- Let's try to find index of 45 in the earlier array if we sort it.

# Binary Search 💡

Search a sorted array by repeatedly dividing the search space in half. Begin with a space covering the whole array. If the value of the target is less than the item in the middle of the interval, narrow the interval to the lower half. Otherwise, narrow it to the upper half. Repeatedly check until the value is found or the space is empty. Works in logarithmic time.
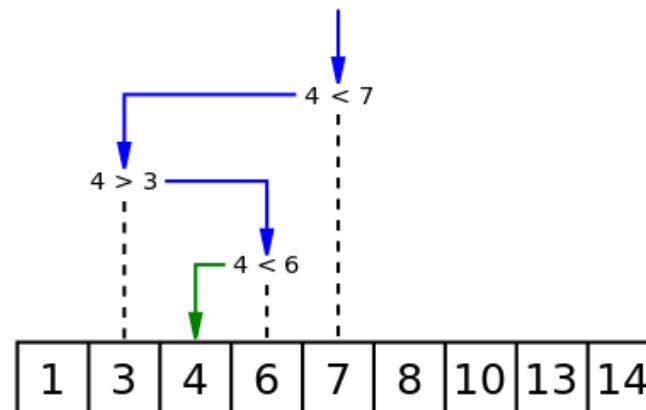
# Template for Binary Search 💡

```
l=0, r=n, ans=0
while(l<=r):
    mid = l+(r-l)/2
    if(check(mid)):
        ans = mid
        l = mid+1
    else:
        r = mid-1;
return ans
```

What is this check ?

# STL Binary Search ☼

binary_search (a , a+n , target) : return true if value is present in array        otherwise false.

lower_bound (a , a+n , target) : returns the smallest value >= target

upper_bound (a, a+n , target) : returns the smallest value > target
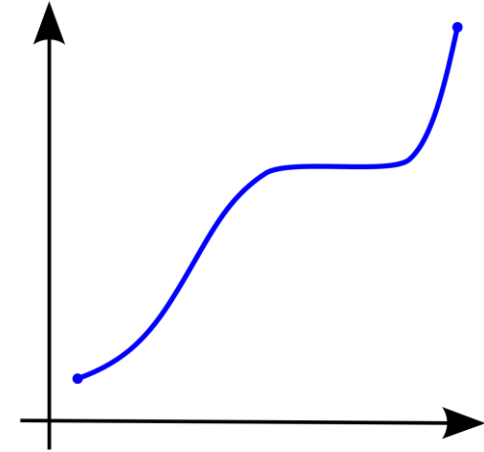
# Time Complexity Discussion ⚙

We are reducing our search space at every step into half of current search space.

What is the recurrence we are using ?

# Requirement to use BS 💡

## Monotonicity

▸ f(x) > f(y) iff x > y (increasing monotonic)

▸ f(x) < f(y) iff x > y (decreasing monotonic)

# Some Problems

1) Find index/position of first element >=x in sorted array.

| 1 | 2 | 5 | 6 | 10 | 11 | 20 | 31 |

Number of elements greater than 2 are ?

# Some Problems

2) Number of elements in the range of *l* to *r* in Sorted Array.

| 1 | 2 | 5 | 6 | 10 | 11 | 20 | 31 |
|---|---|---|---|----|----|----|----|

Number of elements in range of 4 to 12 are ?

# Some Problems

3) Find sqrt(x) in log(x) without using inbuilt sqrt() function.

(Note: Given x > 0 and square root is an Integer)

We don't have an array here!!  XD.

Hint: Think about Monotonicity

# Binary Search on answer ☀

➢ Consider a predicate P defined over some ordered set S (the search space). The search space consists of candidate answers to the problem. In our case, a predicate is a function which returns TRUE or FALSE. We use the predicate to verify if a candidate answer is legal or not.

➢ Example: We have the set of numbers {1, 2, 3, 4, 5}. Our predicate function could be following: Return TRUE if the number is less than 3 and FALSE otherwise.

Now, if we pass 2 to this function, it will return TRUE right?

# Predicate Function ☀

Condition on Predicate Required for doing Binary Search on Answer

P(X) is True => P(Y) is True for all Y > X
&&
P(X) is False => P(Y) is False for all Y < X

| F | F | F | F | T | T | T | T | T | T | T |
|---|---|---|---|---|---|---|---|---|---|---|

P(X) is True => P(Y) is True for all Y < X
&&
P(X) is False => P(Y) is False for all Y > X

| T | T | T | T | T | F | F | F | F | F | F |
|---|---|---|---|---|---|---|---|---|---|---|

Problem : Check your understanding of predicate function