$$\Rightarrow O\left(\frac{\Sigma \text{all possible runtimes}}{\text{total no of possibilities}}\right)$$

## Best worst and Expected Case

Sometimes we get lucky in life.. Exams cancelld when you were not prepared, Surprise test when you were prepired etc ⇒ Best Case

Some times we get unlucky Qustions you never prepared asked in exams, rain during Sports period etc ⇒ worst Case

But Overall the life remains balance withe mixture of lucky and unlucky times ⇒ Expected Case

Analysis of a search algorithm
Consider an array which is Sorted in increasing Order

| 1 | 7 | 8 | 28 | 50 | 180 |
|---|---|---|---|---|---|

we have to Search a given number in this array and report whether its present in an array or not.

Algo 1 → Start from first element until an element greater than or Equal to the number to be searched is found

Algo 2 → check whether the first or the last element is equal to the number if not find the number between these two elements (Center of the array) If the Center element is greater than the number to be Searched, repeat the process for first half else repeat for Second half until number is found.

Analyzing Algo 1

If we really get lucky, the first element of the array might turn out to be the element we are searching for. Hence we made just one comparison

Best Case Complexity = O(1)

If we are really unlucky the element we are searching for might be the last one

worst case complexity = O(n)

For calculating Average Case time, we sum the list of all the possible cases runtime and divide it with the total number of cases

$\Downarrow$

Sometimes calculation of average case time gets very complicated

Analyzing Algo 2

If we get really lucky the first element will be the only one which gets compared

Best case Complexity = O(1)

If we get unlucky we will have to keep dividing the array into halves until we get a single element (The array gets finished)

worst case Complexity = O(log n)
what log(n)? what is that?

log(n) → Number of times you need to half the array of Size n before it gets exhausted

$$\log 8 = 3 \Rightarrow \overset{1}{\frac{8}{2}} \rightarrow \overset{+1}{\frac{4}{2}} \rightarrow \overset{+1}{\frac{2}{2}} \rightarrow \text{Can't break anymore}$$

$$\log 4 = 2 \Rightarrow \underset{1}{\frac{4}{2}} \rightarrow \underset{+1}{\frac{2}{2}} \rightarrow \text{Can't break anymore}$$

log n Simply means how many times I need to divide n units such that we cannot divide them (into halves) anymore.

Space Complexity
Time is not the only thing we worry about while analyzing algorithms. Space is Equally important

Creating an array of Size n → O(n) Space
↳ Size of input

If a function Calls itself recursively n times its Space Complexity is O(n)

why Cant we calculate Complexity in Seconds?
→ Not everyone's Computer is Equally powerful.
→ Asymptotic analysis is the measure of how time (runtime) grows with input.