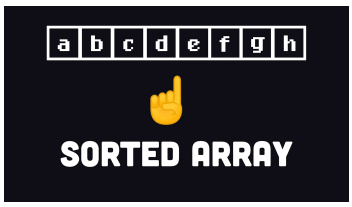
Binary Search

Tags	Algorithm Divide&Conquer MyNotes Searching
Companies	EVERY
Created time	@October 30, 2022 2:52 PM
Last edited time	@October 31, 2022 12:03 PM
URL	

ℹ️ Definition - Binary Search is a optimal searching algorithm that works with sorted input.

Why it is called Binary Search?



Because it uses a approach where it looks for the key/element in the MIDDLE of the sorted list and split it into two this is why its called the binary search. this method is also know as Divide&Conquer method.

why do we need this? motivation behind this? How it is Different from the regular search?

Given a list to search a particular element in it if you look each element one by one linearly it doesn't matter you start front or back the element can be anywhere first

middle near end or exactly end. in smaller queries/list it doesn't really matter but as the input grows searching linearly can take a significant time.

Ex: suppose you take 1 second for one element and you are given a list one 1 million elements. linearly it might take you 11 days to find the element but if the list is sorted you can apply binary search and get the answer in under 20 second. this is the power of logarithmic approach.

How this works?

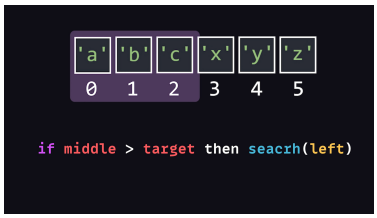
starting from 0th element.

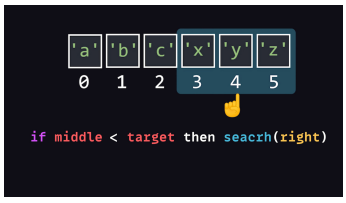
starting from middle element.

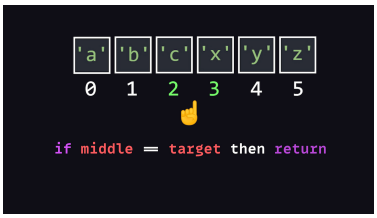
when you looking in any ascending sorted list it is always the case that the elements after that will be bigger and before that will be smaller binary search take advantage of that fact and instead of starting from the first element it start the search from the middle of the array.

if the element we are looking is smaller than the middle element than the elements after that are of no use as they will be bigger than middle hence bigger than the target element similarly if middle element is smaller than target elements before that will be of no use.

so what we can coding wise is take a three variable one from zero to iterate from start and other size-1 to iterate from last element then calculate mid and set it to third element and start iterating in a loop in each iteration we will calculate middle if element on middle index is greater than target we will simply set back as mid-1 and start search again if middle is smaller we will set start var to mid+1 and repeat the process until the middle element is equal to target or if there is no element left in between. since we are increasing the start and decreasing the end there might be a situation when start is bigger than back which indicates that there is no element left in between and if target is not found yet than the element is not present in that list. so we end the loop with this condition and return -1 or throw an exception regarding the application.

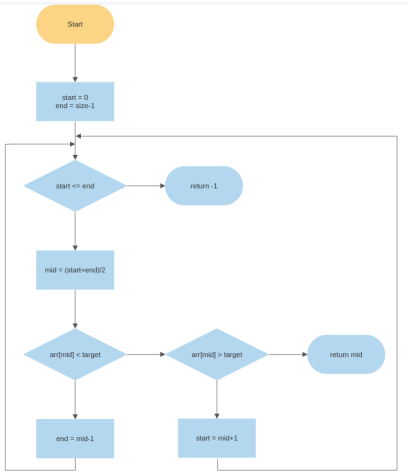






lets write pseudo code

```
int findIndexOf(int arr[],int size,int target)
{
  int start = 0;
  int end = 0;
  while(start<=end)
  {
    int mid = (start+end)/2;
    if(arr[mid] > target) end = mid-1;
    else if(arr[mid] < target) start = mid+1;
    else return mid;
  }
  return -1;//not found
}
```

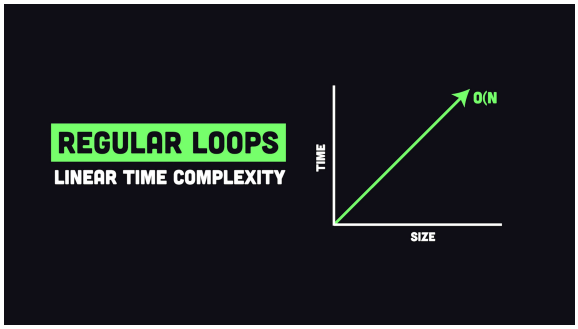


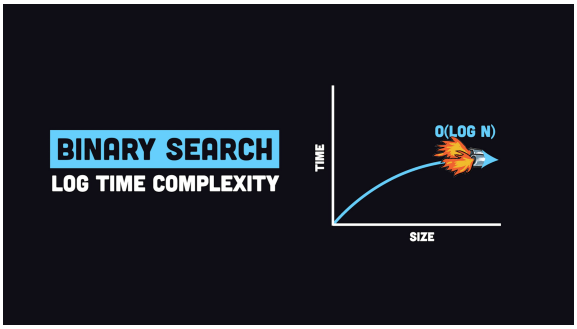
time and space complexity

So we have seen and proved that binary searched is faster than linear search but how fast it is actually lets estimate

giving list of x it will divide it into two repeatedly util it x = 1 if it doesn't sound familiar lets take example values for x.

```
if x = 2 :
itr 1 => 2/2 = 1
so when x is 2 it will get answer in one iteration
now lets take more and bigger example
x = 32 :
itr 1 => 32/2 = 16
itr 2 => 16/2 = 8
itr 3 => 8/2 = 4
itr 4 => 4/2 = 2
itr 5 => 2/2 = 1
in every iteration we make x half of itself until it reachs 1
it takes 5 iteration to divide 32 to 2 till we get it to 1.
or we can say it takes 5 iteration for 2 to double itself till it become 32
we can clearly see the relation as 2 raised to power 5 is 32 actualy,
which is very obvious since we are just doubling 2
we can write it mathematicly as
2^5=32 but we have to write it in terms of N in this case N = 32 so we can write it as
log(32) = 5 it represent it takes 5 iteration to execute binary search completly
from this we can tell taking log to n will get as to the number of iteration we want
long(N) = number of iteration
```





▼ src

Binary Search Algorithm in 100 Seconds

Binary Search is an algorithm that can find the index of an element in a sorted array data structure. You've likely used Binary Search it in everyday life without even realizing it.

https://www.youtube.com/watch?v=MFhxShGxHWc&t=4s