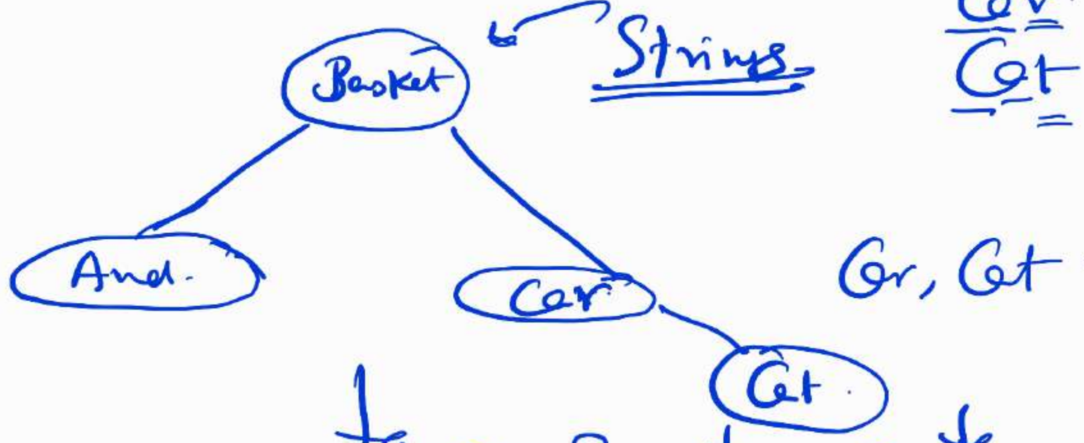
BST



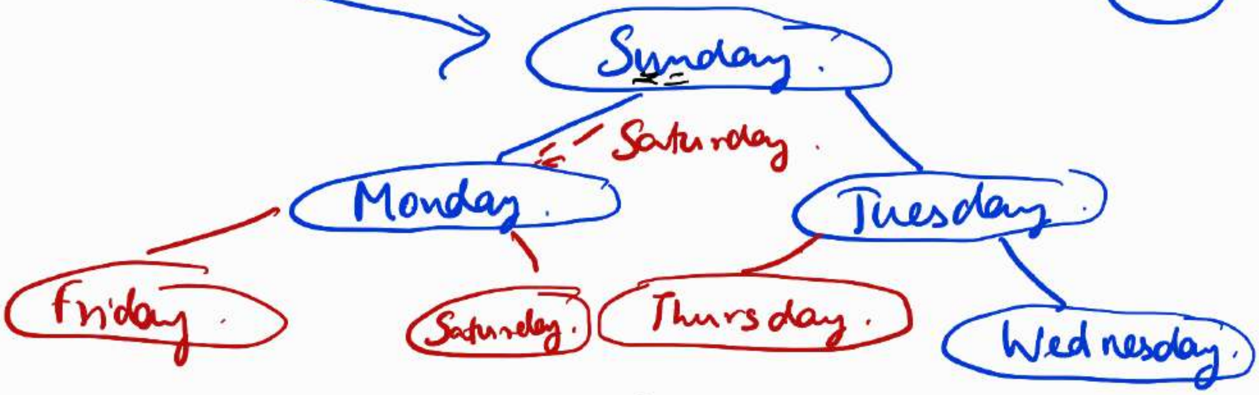
Inorder BST:-

Author & Name & Year

And Basket Car Cat

→ Sunday, Monday, Tuesday, Wed, Thursday, Friday, Saturday.

1 2 3 4 5 6 7



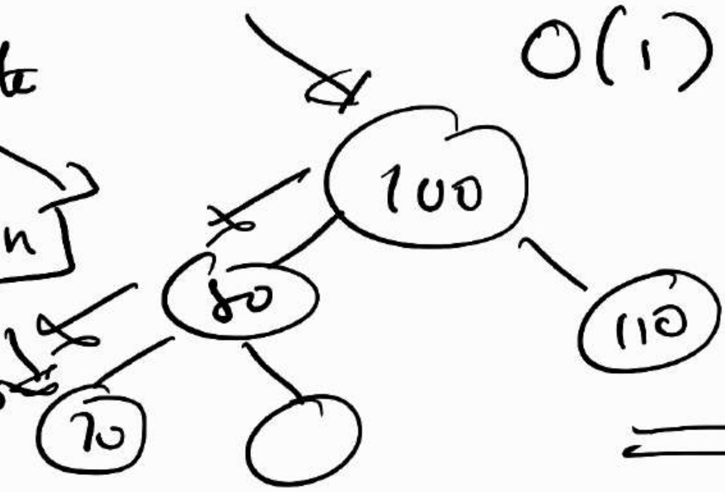
BST

log n

Friday Monday Saturday Sunday Thursday Tuesday Wednesday

Complete

Worst $\lceil \log n \rceil$



O(1)

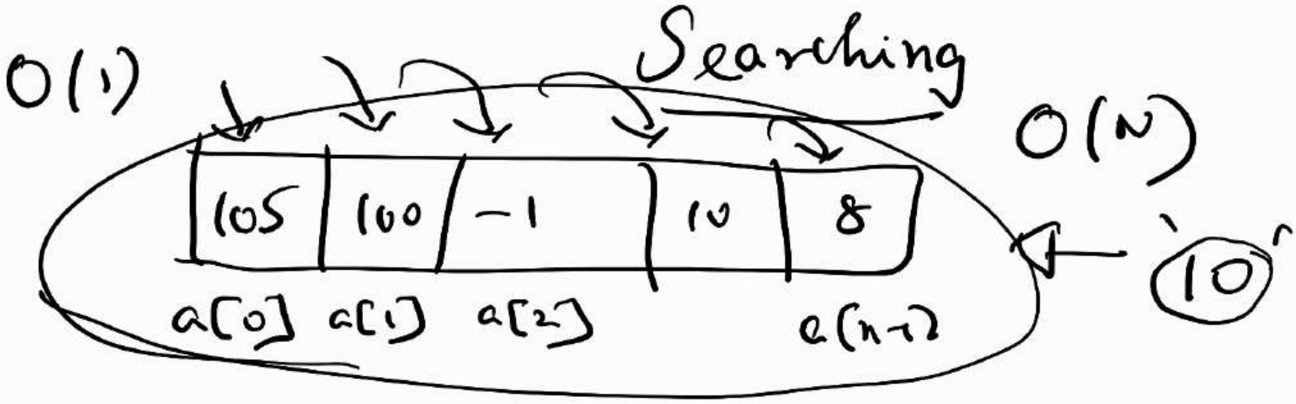
$n \times \log n$

+

$O(n) \to c.n$

$\Longrightarrow (n \log n)$

Searching



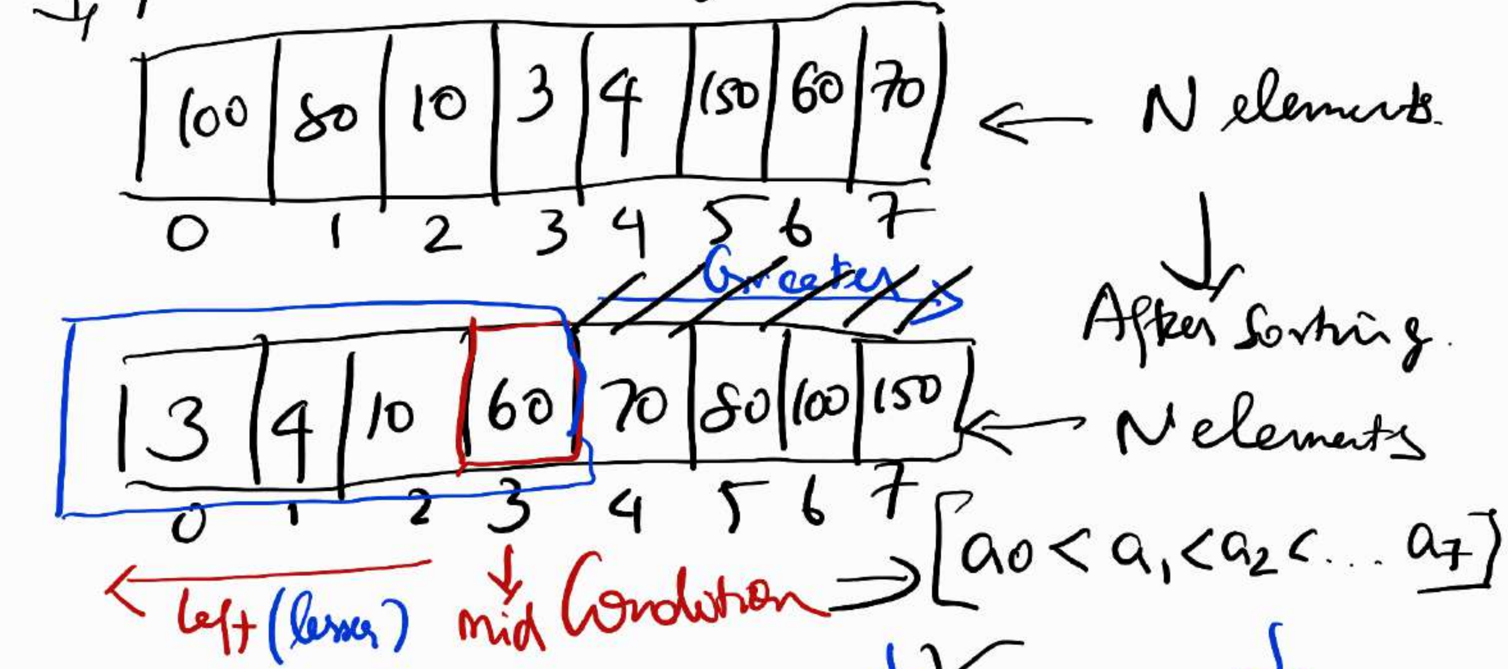
Best Case - O(1)

Worst Case - O(N)

$\boxed{O(\log n)}$ → Complete BST

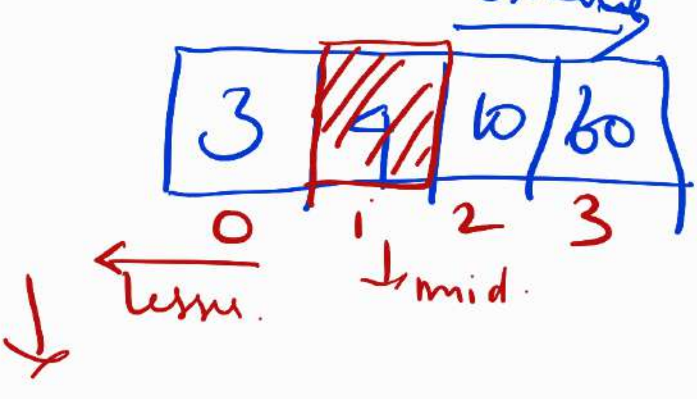
Binary Search.

I/p.



I/p :- $\boxed{4 < 60}$ (or) 4 > 60

n → n/2



3

4 < 4 (or) 4 > 4

(or) $\boxed{4 = 4}$

```
void main (      )
{  Scanf ("The Key to be Searched %d", &Key);
 int Search_index = BinarySearch (Arr, low, high,
                                       Key);
        if  (Search_index == -1)
      {     pf("Element not found");
      }
  else
      {  pf(" Element present at index %d",
                              search_index);
  }   }
```

```
int BinarySearch(Arr, low, high, Key)
{
    if ( low <= high)
    {
      int mid = ⌊(low + high)/2⌋;
      {  if ( arr[mid] == key)  }
         {                          } → Element
           return mid;              }   is present(or)not
         }
      else if ( arr[mid] < Key)
         {
            return BinarySearch(Arr, mid+1, high,
                                 Key);
      else if (arr [mid] > Key)
         {
            return BinarySearch(Arr, low, mid-1,
                                 Key);
         }
    }

      pf (" Element is not fond");
      return -1;
}
```

mid
high
left
main
mid =

$\boxed{O(\log n)}$