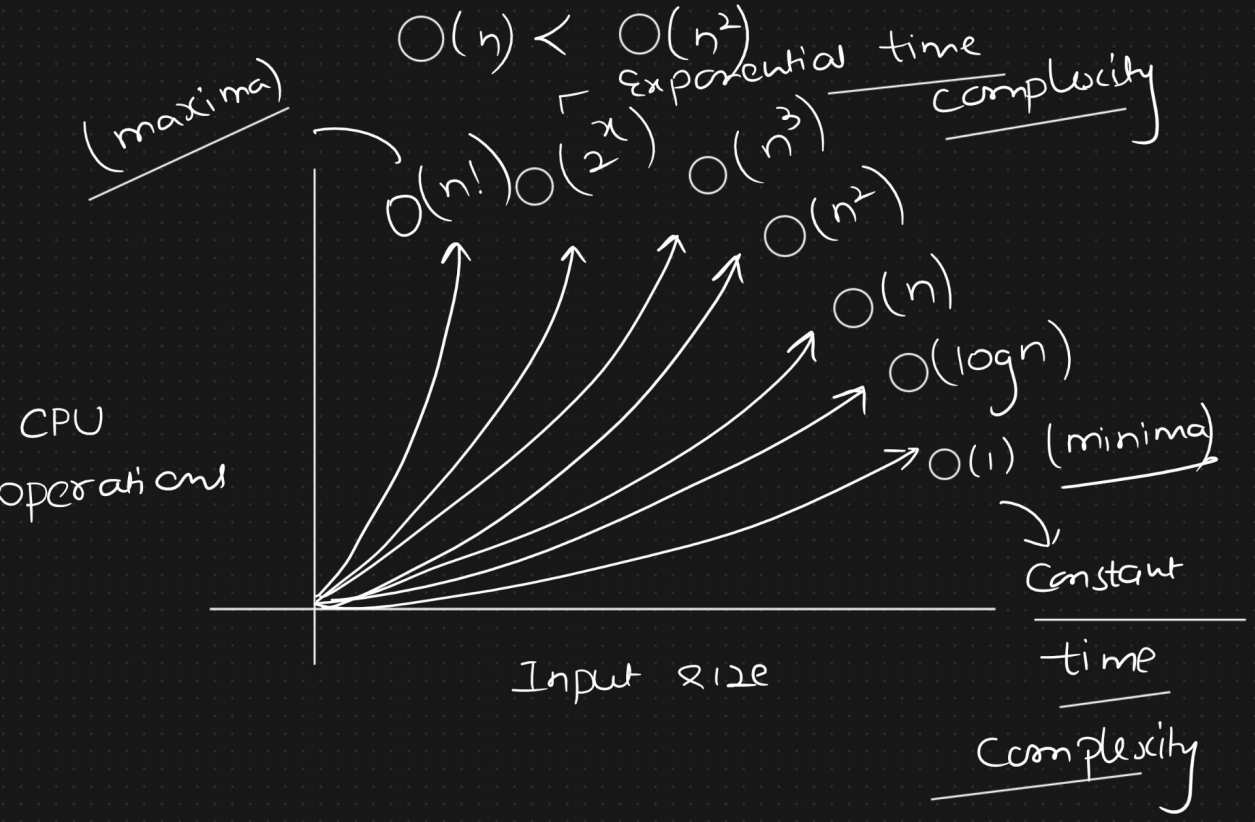


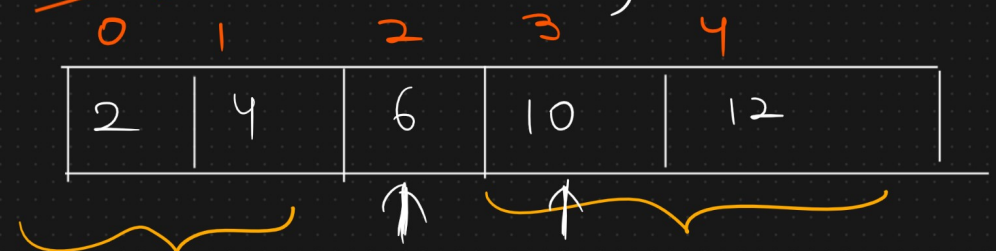
Example 3

↳ Array Data Structure

```
n ⟶ array size
binarySearch(int[] arr, int x){
    low = 0;
    high = n-1;
    while(low <= high){
        mid = (low + high)/2;
        if(arr(mid) == x){
            return mid;
        }
        else if(arr(mid) < x){
            low = mid+1;
        }
```

```
else {
    high = mid - 1;
}
return -1;
}
```

Array → Sorted array



Array(3) = 10

x = 10

output = 3

x = 13

output = -1

low = 0

high = 4

mid = 2

---

arr(2) = 6 == 10 X

$6 < 10$

↳ right side of 6

low = 3

$$mid = \frac{3+4}{2} = 3$$

n ——— 0th iteration

$n/2$ ——— 1st iteration

$n/2^2$ ——— 2nd iteration

↓ k iterations

$$\frac{n}{2^k} = 1$$ ← Searching element found

$$n = 2^k$$

$$\log_2 n = k \log_2 2$$

$$k = \log_2 n$$ ——— # iterations

Time complexity = $O(\log_2 n)$