· Binary Search

Pseudo Code :-

```
Binary Search (A, Key):
    first = 1
    last = length (A)
    flag = 0
    while first <= last and flag == 0
        mid = (first + last) // 2
        If A[mid] == Key
            flag = 1
    elseIf Key < A[mid]
            last = mid - 1
        else
            first = mid + 1
    return flag
```


Scanned with CamScanner

• **DRY RUNNING**

BEST CASE :-

A = [1, 2, 3, 4, 5, 6, 7, 8]

Key = 4

• In binary search, when key is present at mid then it is best case because it takes only one comparison to find the key

1. first = 1

2. last = 8
3. flag = 0   1st iteration of while
4. while 1 <= 8 and 0 == 0
5. mid = (1+8)//2 = 4
6. if A[4] == 4 → 1
   4 == 4

7. flag = 1
8. ______
9. ______
10. ______
11. ______

2nd iteration of while

4. while 1 <= 8 and 1 == 0 → false
   loop terminates

12. return 1

Line no of code	Time/instruction	fequency
1	C	1
2	C	1
3	C	1
4	C	1+1=2
5	C	1
6	C	1
7	C	1
8	C	0
9	C	0
10	C	0
11	C	0
12	C	1

$T(n) = 1+1+1+2+1+1+1+1$

$T(n) = 9$

$T(n) = K$

• DISCUSSION:-

- Best case time complexity of binary search is O(1)

- means that, only takes one comparison to find the key

- Binary search in best case requires constant time to execute

Time Complexity:-

Key = 10

- First iteration

1	2	3	4	5	6	7	8

first mid last

mid = first + last/2

$mid = \frac{1+8}{2} = 4$

- since A[mid] < Key → line no 10

4 < 10

11. first = mid+1

→ first = 5

- 2nd iteration

5	6	7	8

first mid last

→ Key = 10

first = 5

last = 8

$mid = \frac{5+8}{2} = 6$

10. since A[mid] < Key

6 < 10

11. first = mid + 1

→ 7

- 3rd iteration:-

7	8

first mid last

→ Key = 10

first = 7

last = 8

$mid = \frac{7+8}{2} = \frac{15}{2} = 7$

7 < 10

10. A[mid] < Key

11. first = mid+1

→ 7+1 = 8

- 4th iteration

8

Scanned with CamScanner

1st iteration: $\{1,2,3,4,5,6,7,8\} \rightarrow n \rightarrow \frac{n}{2^0} = n$

2nd iteration: $\{5,6,7,8\} \rightarrow n/2 \rightarrow \frac{n}{2^1} = \frac{n}{2}$

3rd iteration: $\{7,8\} \rightarrow n/4 \rightarrow \frac{n}{2^2} = \frac{n}{4}$

fourth iteration: $8 \rightarrow n/8 \rightarrow \frac{n}{2^3} = \frac{n}{8}$

5th iteration: doesn't run

In general terms

$\frac{n}{2^k} = 1 \rightarrow$ To prove logarithmic growth

2^k ↓ amount of steps to make n=1

8, 4, 2, 1 → Indicates logarithmic growth

• If $n = 16$

$\frac{16}{2^k} = 1$

$16 = 2^k$

$2^4 = 2^k$

$\boxed{k=4} \rightarrow \boxed{n=16}$

• If $n = 4$

$\frac{4}{2^k} = 1$

$4 = 2^k$

$4 = \cancel{2^2}\, 2^k$

$2^2 = 2^k$

$\boxed{k=2} \rightarrow n=4$

• If $n = 8$

$\frac{8}{2^k} = 1$

$8 = 2^k$

$2^3 = 2^k$

$\boxed{k=3} \rightarrow n=8$

Scanned with CamScanner

Date

Line no of code	Time / instruction	frequency
4	C	$5 = \log_2(n)+2 = \log_2(8)+2 = 5$ ↳ express in terms of log because we prove that the growth is logarithmic
5	C	$4 = \log_2(n)+1$
6	C	$4 = \log_2(n)+1$
7	C	0
8	C	$4 = \log_2(n)+1$
9	C	0
10	C	$4 = \log_2(n)+1$
11	C	$4 = \log_2(n)+1$
12	C	1

$$\boxed{t(n) = K\log_2(n)+K}$$

↓

$$T(n) = \log_2(n)+2 + \log_2(n)+1 + \log_2(n)+1 + \log_2(n)+1 + \log_2(n) + \log_2(n)+1$$

$$T(n) = 6\log_2(n)+6$$

$$\boxed{T(n) = K\log_2(n)+K}$$

Date

• DISCUSSION:-

- Worst case complexity of binary search is O(logn)
- binary search in worst case grows logarithimically.