CMPS 10 10/26/10

## Binary Search

```
1.)  L = 1
2.)  R = n
3.)  found = false
4.)  while L ≤ R and not found
5.)      m = ⌊(L+R)/2⌋
6.)      if target == a_m
7.)          found = true
8.)      else if target < a_m
9.)          R = m - 1
10.)     else
11.)         L = m + 1
12.) if not found
13.)     m = 0
14.) print m
15.) stop
```

Ex. target = 8 , n = 10

|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| a | 2 | 3 | 4 | 7 | 10 | 11 | 15 | 20 | 25 | 28 |

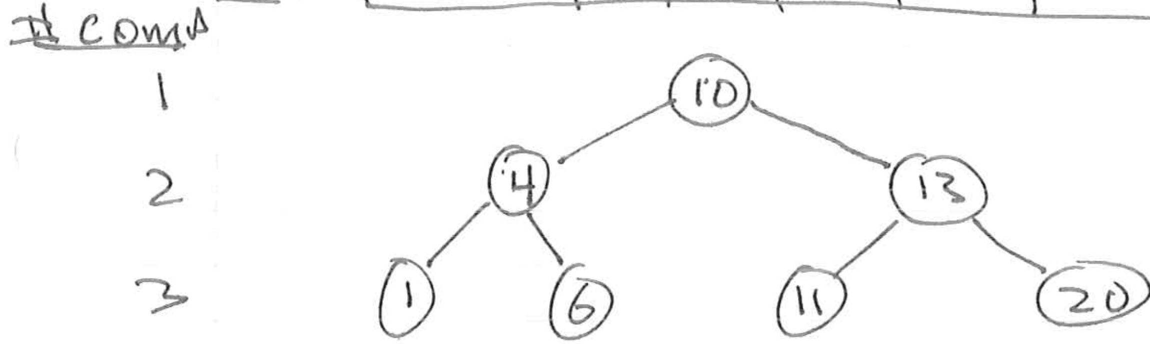
| L | R | m | found |
|---|---|---|---|
| ~~1~~ | ~~10~~ | ~~5~~ | F |
| ~~3~~ | 4 | ~~2~~ | |
| ~~4~~ | | ~~3~~ | |
| 5 | | ~~4~~ | |
| | | $\boxed{0}$ ← Print | |

# Run Time of Binary Search.

Basic Op.: comparison of target to a list element.

Ex.

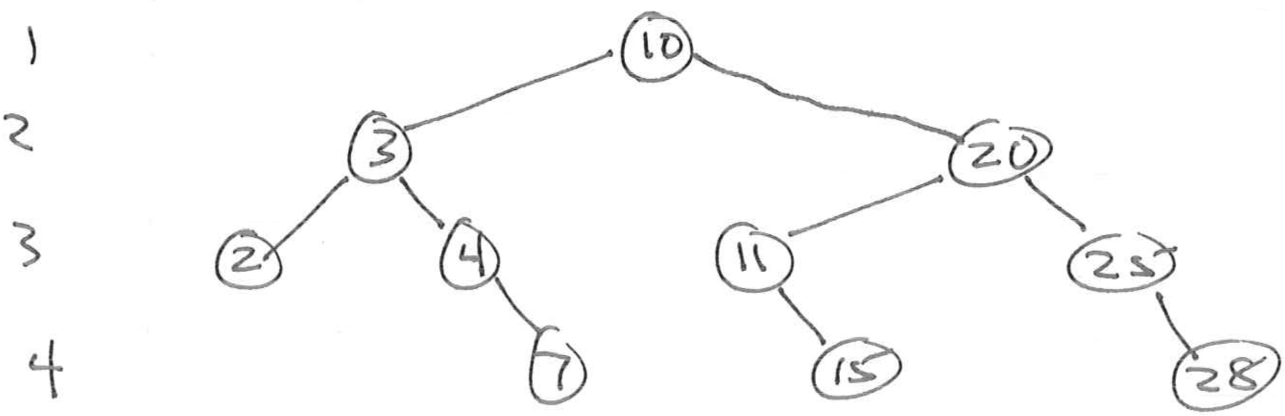
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|
| 1 | 4 | 6 | 10 | 11 | 13 | 20 |




Worst case # of comparisons in this list = 3

Ex. $n = 10$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| # comp | 2 | 3 | 4 | 7 | 10 | 11 | 15 | 20 | 25 | 28 |




- worst case # comp $= 4$
- avg case # comp (assume target in list, equally likely in any pos.)

$$\frac{1\cdot 1 + 2\cdot 2 + 4\cdot 3 + 3\cdot 4}{10} = \frac{29}{10} = 2.9$$

Let $W(n)$ = worst case # of comparisons by B.S. on lists of length $n$.

<u>Goal</u>: find a general formula for $W(n)$.

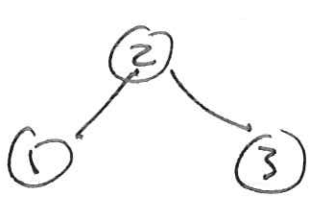
<u>Note</u>: $W(n)$ depends only on the list length $n$, not list contents. so $W(10)=4$
$W(7)=3$

Standard list of length $n$

1 2 3 . . . . . . $n$

<u>Ex</u> $n=3$.

1 2 3




| | #comp |
|---|---|
| | 1 |
| | 2 |

$W(3)=2$

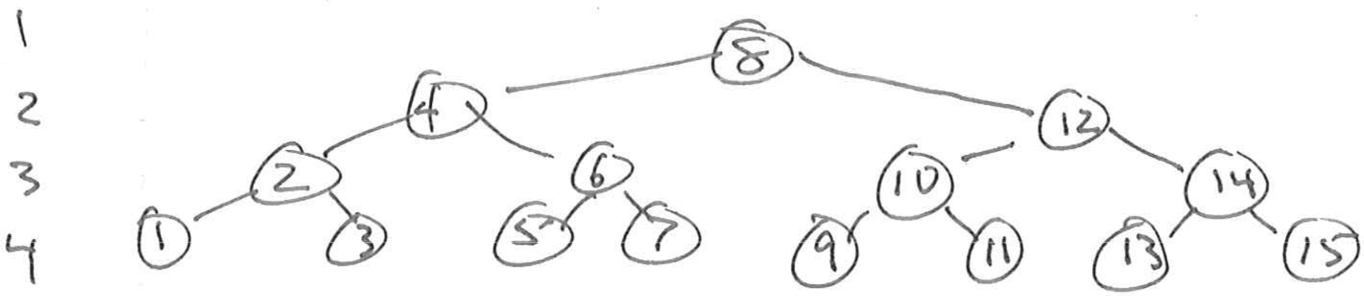
Ex $n = 7 = 2^{3} - 1$

1 2 3 4 5 6 7


$w(7) = 3$

Ex, $n = 15 = 2^{4} - 1$

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15




$w(15) = 4$

Observe: $w(31) = 5 = 2^{5} - 1$

$w(63) = 6 = 2^{6} - 1$

$w(127) = 7 = 2^{7} - 1$

Observe:

If

$$2^{k-1} - 1 < n \le 2^{k} - 1$$

Then

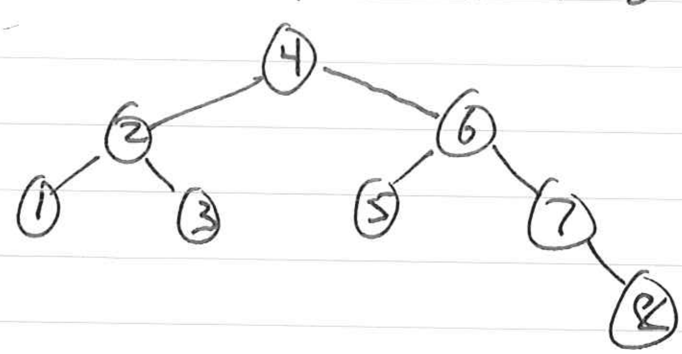
$$W(n) = k$$

Exercise: Draw Binary Search trees for intermediate values of $n$. i.e.

$n = 1, 2, \cdot, 4, 5, 6, \cdot, 8, 9, 10, 11, 12, 13,$

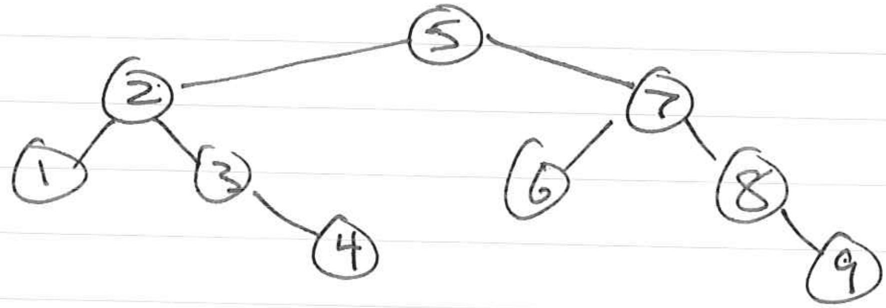
$14, \cdot, 16, 17$

Ex $n=8$: 1 2 3 4 5 6 7 8



Ex $n = 9$

1 2 3 4 5 6 7 8 9 # comp



1
2
3
4

## Logarithms

Let $b > 1$, $x > 0$. Then $\log_b(x)$ is defined to be the power you must raise $b$ to to get $x$.

$$y = \log_b(x) \quad \text{iff} \quad x = b^y$$

Ex. $\log_3 9 = 2$ since $3^2 = 9$

$\log_5(125) = 3$ since $5^3 = 125$

$\log_{10}(10000) = 4$ since $10^4 = 10\,000$

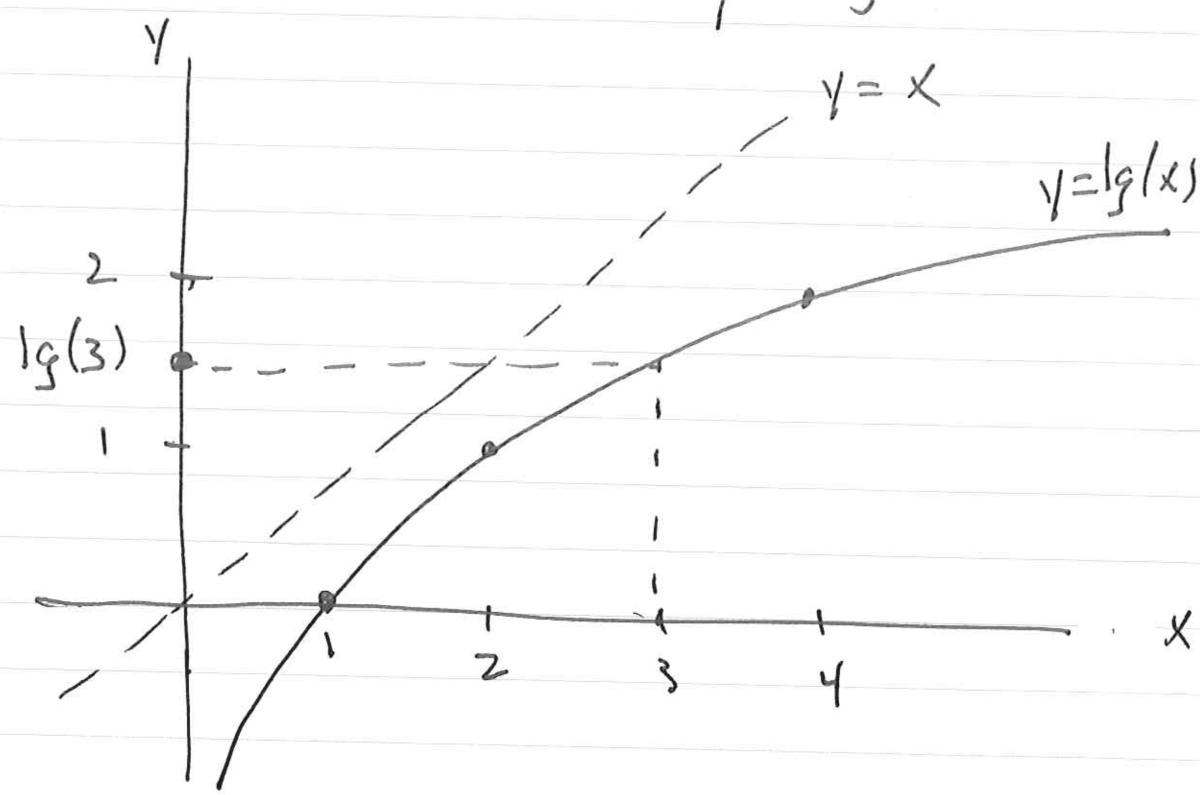
$\log_2(32) = 5$ since $2^5 = 32$

Special notation

$b = 10$ : log  common log / Naperian log

$b = e = 2.7182\ldots$ : ln  natural log

$b = 2$ : lg  Binary log



**note** $\lg(n)$ grows slower than $n$

In fact $\frac{\lg(n)}{n} \longrightarrow 0$ as $n \to \infty$

**Recall!**

$w(n) = k$ iff $\underbrace{2^{k-1} - 1 < n \leq 2^k - 1}$

$\therefore \quad 2^{k-1} < n+1 \leq 2^k$

$\therefore \quad 2^{k-1} \leq n < 2^k$

$\therefore \quad \lg(2^{k-1}) \leq \lg(n) < \lg(2^k)$

$\therefore \quad k-1 \leq \lg(n) < k$

$\therefore \quad k-1 = \lfloor \lg(n) \rfloor$

$\therefore \quad k = \lfloor \lg n \rfloor + 1$

Thm

$$W(n) = \lfloor \lg(n) \rfloor + 1$$

notice $\lfloor f(n) \rfloor = \Theta(f(n))$

Conclude

$$W(n) = \Theta(\lg(n)).$$

Example

- unsorted list length $n$, 1 target

Cost of sorting: $\Theta(n^2)$

Cost of searching: $\Theta(\log_2(n))$

Cost of both: $\Theta(n^2 + \lg(n)) = \Theta(n^2)$

cost of seq. search $= \Theta(n)$

Ex unsorted list length n
many targets

Sort first
Search cheaply