(continued)

noon # Binary Search

1 pm
- Optimised way of searching in array

2 pm
- Arrays need to be sorted

3 pm

in Linear search : N => No. of elements max. comparisn

4 pm

5 pm arr: [2, 4, 6, 9, 11, 12, 14, 20, 36, 48] target : 36
(s, mid, s_1, m_1, s_1, e)

evening step-1 : Find middle element

7 pm step-2 : If target > mid => search in right

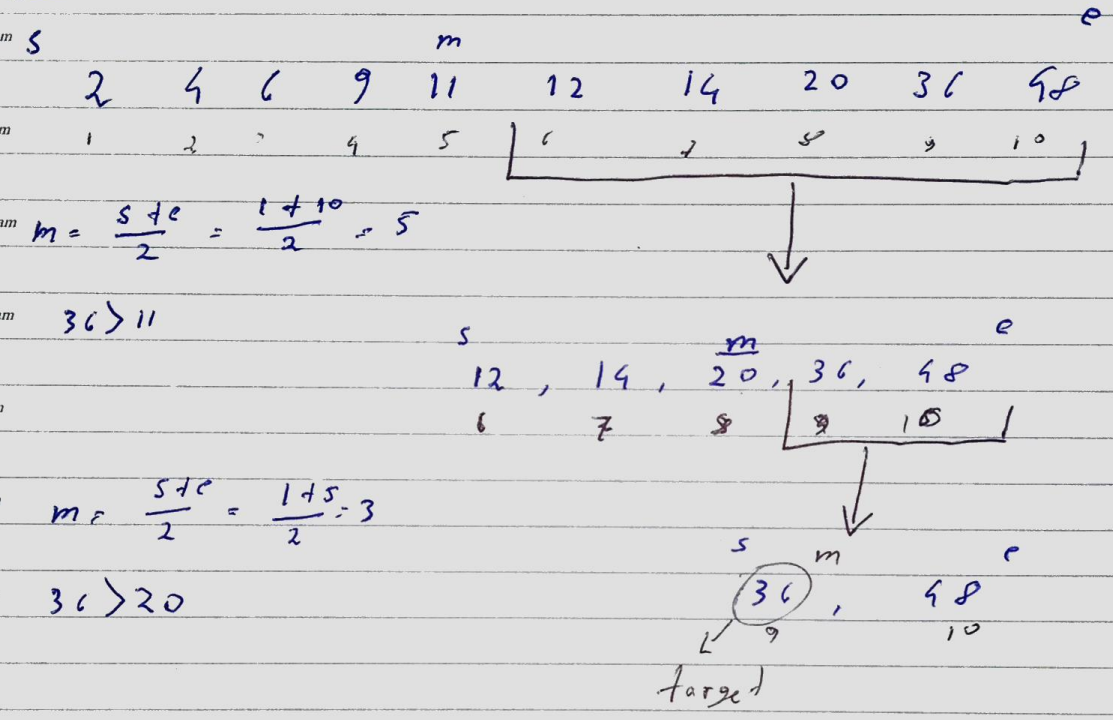
else search in left

3 : If middle == target, answer

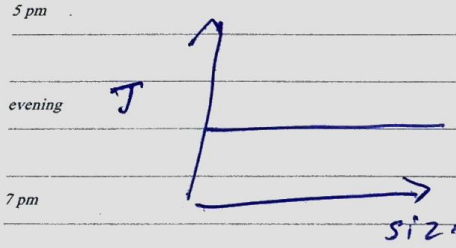
4 : If start > end : element not found

(A) => we are not reaching mid array

Dry run

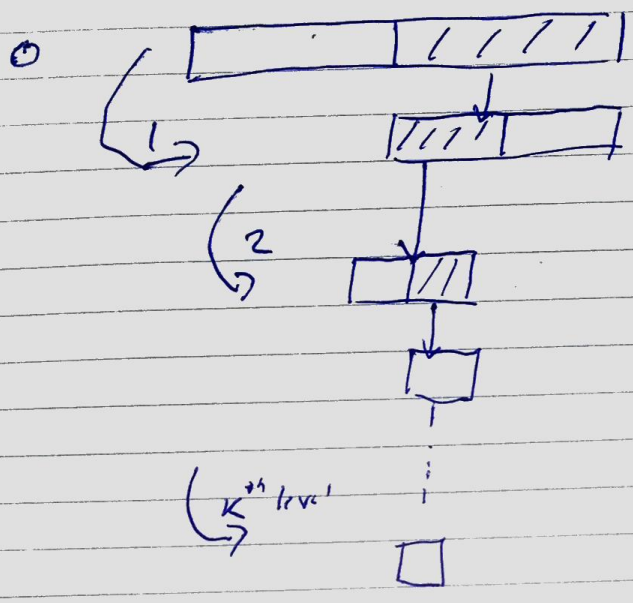


Comparison in best case:



O(1)

Worst case



$N = \frac{N}{2^0}$

$\frac{N}{2} = \frac{N}{2^1}$

$\frac{N/2}{2} = \frac{N}{4} = \frac{N}{2^2}$

$\frac{N}{8} = \frac{N}{2^3}$

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

Also,

$\frac{N}{2^k} = 1 \Rightarrow N = 2^k \Rightarrow \log(N) = \log(2^k)$

$\Rightarrow \log N = k \log 2 \Rightarrow k = \frac{\log N}{\log 2}$

Sunday 24

$k = \log_2 N$

Worst case comparison = $\log_2 N = O(\log N)$

$\therefore$ Search in 1000000 $\Rightarrow$ Linear = 1000000

Binary = 20

Order Agnostic Binary Search :-

8 am

-> If we don't know the order is in assending
9 am or decending but the array is sorted

10 am

Problem

11 am arr : [3, 3, 3, 4, 12, 60, 80, 120, 200]

noon S E

1 pm Compare S and E

2 pm S > E -> ~~assending~~ decending

3 pm E > S -> assending

4 pm