if (target equals contents[mid])

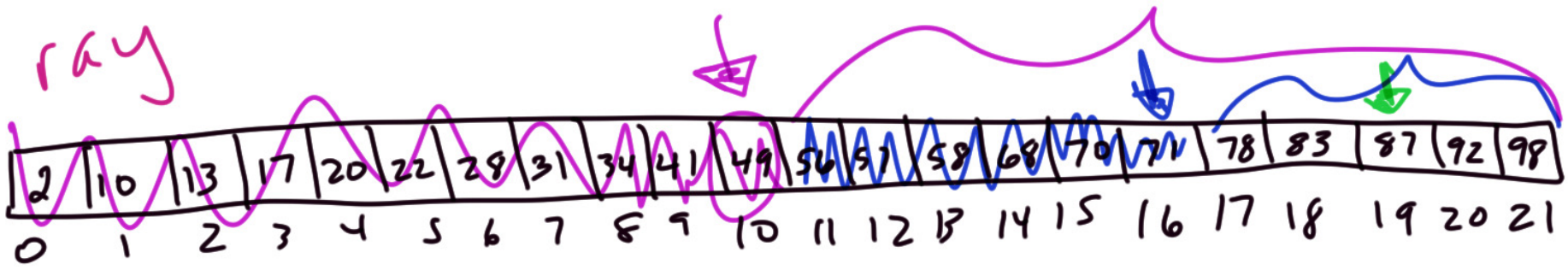
match!

else if (target < contents[mid])

look at the ~~left~~ side

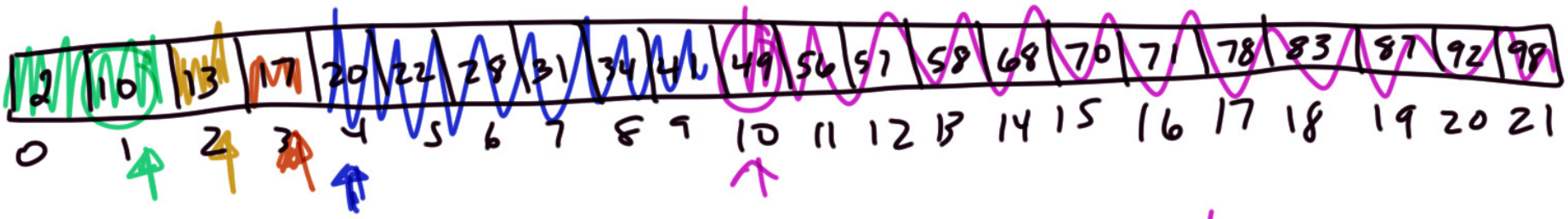
else // target > contents[mid]

look at the right side



target	first	last	mid
87	0	21	(0+21)/2 = 10
87	11 (+1)	21	(11+21)/2 = 16
87	17 (+1)	21	(17+21)/2 = 19

ray:



target	first	last	mid
19	0	21	(0+21)/2 = 10
19	0	9	(0+9)/2 = 4
19	0	3	(0+3)/2 = 1
19	2	3	(2+3)/2 = 2
19	3	3	(3+3)/2 = 3
* 19	4	3	NOT FOUND *

+1

+1

+1

Efficiency

Sequential Search (unsorted data) $O(n)$

Sequential Search (sorted data) $O(n)$

Binary Search (sorted array) $O(\log_2 n)$

16 items

1111

2s

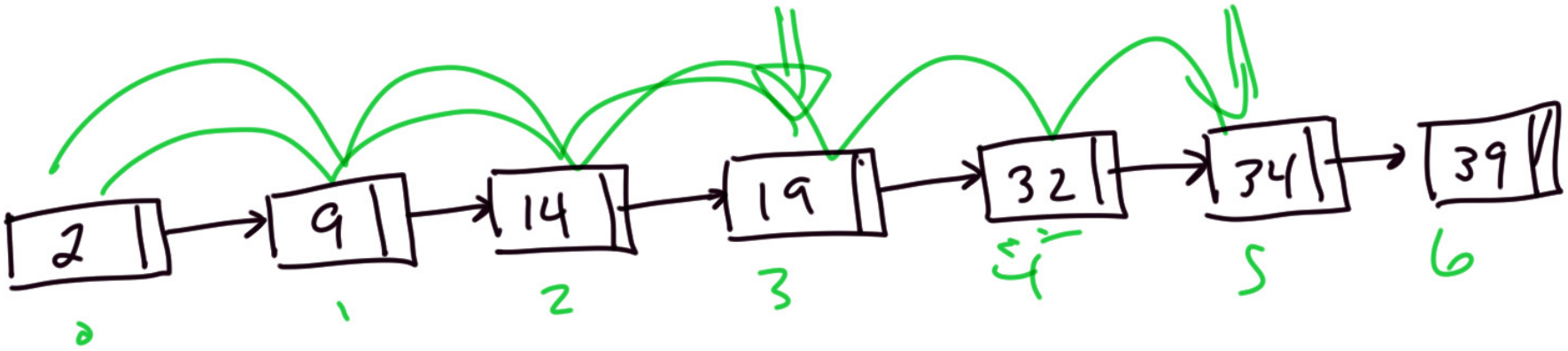
~~1111~~ 111

Iterative vs Recursive Binary Search

Iterative: saves time + memory

Recursive: elegant & easier to understand

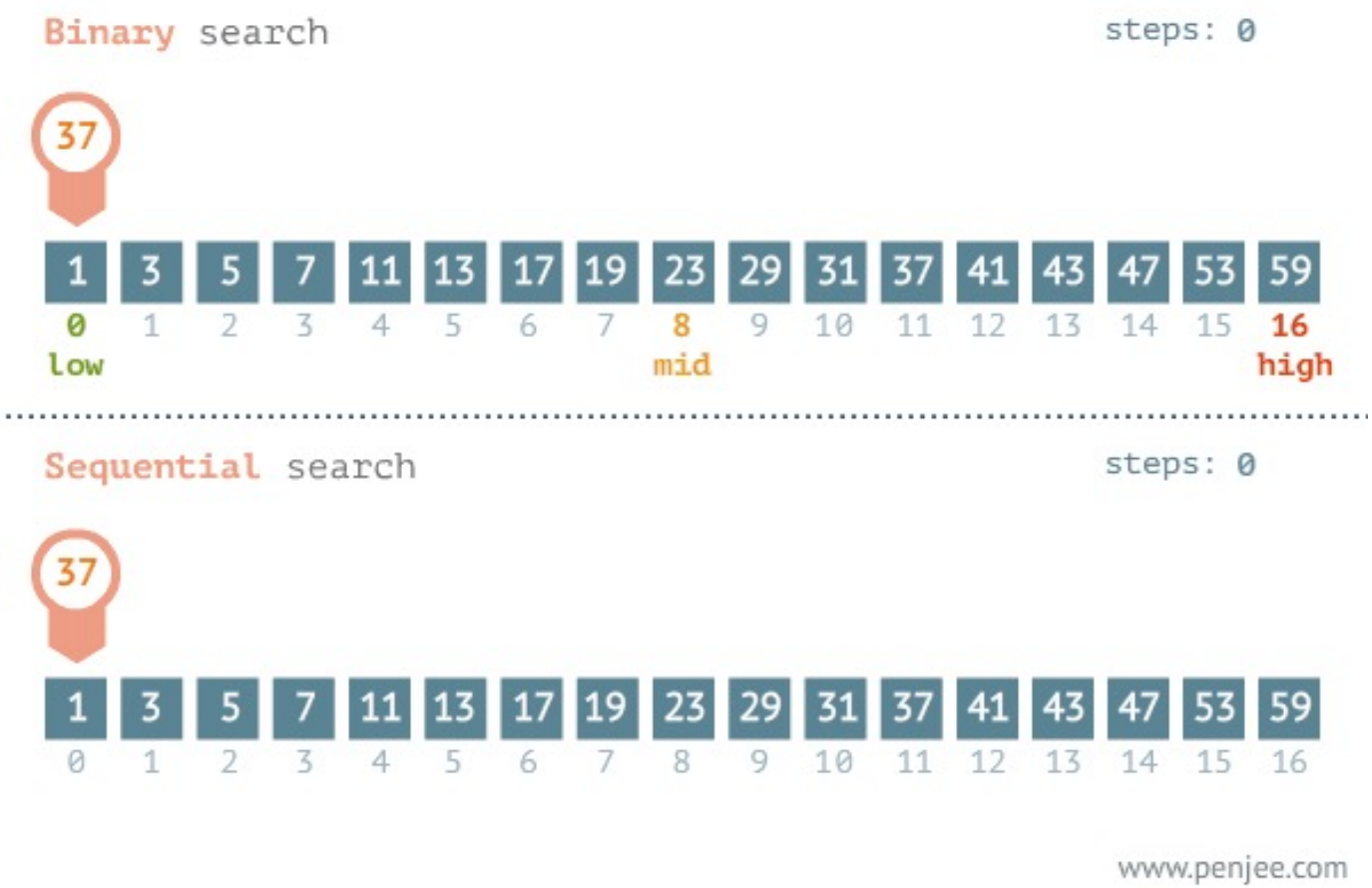
Consider searching a linked-chain implementation



sequential search O(n) * better

binary search

Binary vs sequential search



https://devopedia.org/binary-search