

# Array Algorithms

## Binary Search



# Lecture Contents

- Review Linear Search
- Binary Search
- Readings for IG Computer Science:
  - “*Searching Algorithms: Binary Search*” (top of page 20 until end of page 22), including *Efficiency of Searching Algorithms*



# Linear Search

- Not very efficient
- Frequently used because it's very simple
- Start at the beginning and go through each element step by step

numbers

1	3	5	7	9
0	1	2	3	4

# Binary Search

- Consider... how do you find a specific page in a book?
  - Linear search?
    - Start at page 1 and keep flipping through the pages?





# Binary Search

- Consider... how do you find a specific page in a book?
  - Linear search?
    - Start at page 1 and keep flipping through the pages?
  - Divide and conquer!
    - Open the book somewhere in the middle.
    - Is it the right page?
    - Do we need to search before or after this page?
    - (repeat until found)



# Binary Search

- Things to consider about *binary search*
  - Requires the data to be sorted
    - It takes much longer to sort the data than it does to do a linear search





# Binary Search

- Things to consider about **binary search**
  - Requires the data to be sorted
    - It takes much longer to sort the data than it does to do a linear search
  - The benefits are only significant when the data set is large
    - While the code for **binary search** is slightly more complicated than for **linear search**.





# Binary Search

- We can estimate time by counting the number of comparisons...
  - How long does it take to find an element using *linear search*,
    - worst case?
    - on average?
  - How long does it take to find an element using *binary search*, worst case?

-7	1	4	9	13	21	21	22	32	92
0	1	2	3	4	5	6	7	8	9



# Binary Search

- We can estimate time by counting the number of comparisons...
  - How long does it take to find an element using *linear search*,
    - worst case? **10 comparisons** → for  $n$  elements,  $n$  comparisons
    - on average? For  $n$  elements,  $(n/2)$  comparisons
  - How long does it take to find an element using *binary search*, worst case?

-7	1	4	9	13	21	21	22	32	92
0	1	2	3	4	5	6	7	8	9



# Binary Search

- We can estimate time by counting the number of comparisons...
  - How long does it take to find an element using *linear search*,
    - worst case? **10 comparisons** → for  $n$  elements,  $n$  comparisons
    - on average? For  $n$  elements,  $(n/2)$  comparisons
  - How long does it take to find an element using *binary search*, worst case?
    - $\log_2(n)$

-7	1	4	9	13	21	21	22	32	92
0	1	2	3	4	5	6	7	8	9



# Array Algorithms

## Binary Search