Welcome to this CoGrammar tutorial: Recursion, Sorting and Searching

The session will start shortly...

Questions? Drop them in the chat. We'll have dedicated moderators answering questions.

CoGrammar

Software Engineering Session Housekeeping

- For all **non-academic questions**, please submit a query: www.hyperiondev.com/support
- We would love your **feedback** on lectures: Feedback on Lectures

Software Engineering Session Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and throughout the session, should you wish to ask any follow-up questions.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: Questions

Safeguarding & Welfare

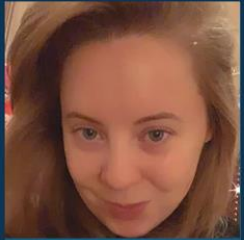
We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Rafiq Manan



Charlotte Witcher

Nurhaan Snyman

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

CoGrammar HyperionDev

Skills Bootcamp Progression Overview

To be eligible for a certificate of completion, students must fulfil three specific criteria. These criteria ensure a high standard of achievement and alignment with the requirements for the successful completion of a Skills Bootcamp.

✓ Criterion 1 - Meeting Initial Requirements

Criterion 1 involves specific achievements within the first two weeks of the program.
To meet this criterion, students need to:

- Attend a minimum of 7-8 hours per week of guided learning (lectures, workshops, or mentor calls) within the initial two-week period, for a total minimum of 15 guided learning hours (GLH), by no later than 15 September 2024.
- Successfully complete the Initial Assessment by the end of the first 14 days, by no later than 15 September 2024.

CoGrammar

Skills Bootcamp Progression Overview

✓ Criterion 2 - Demonstrating Mid-Course Progress

Criterion 2 involves demonstrating meaningful progress through the successful completion of tasks within the first half of the bootcamp.
To meet this criterion, students should:

- Complete 42 guided learning hours and the first half of the assigned tasks by the end of week 7, no later than 20 October 2024.

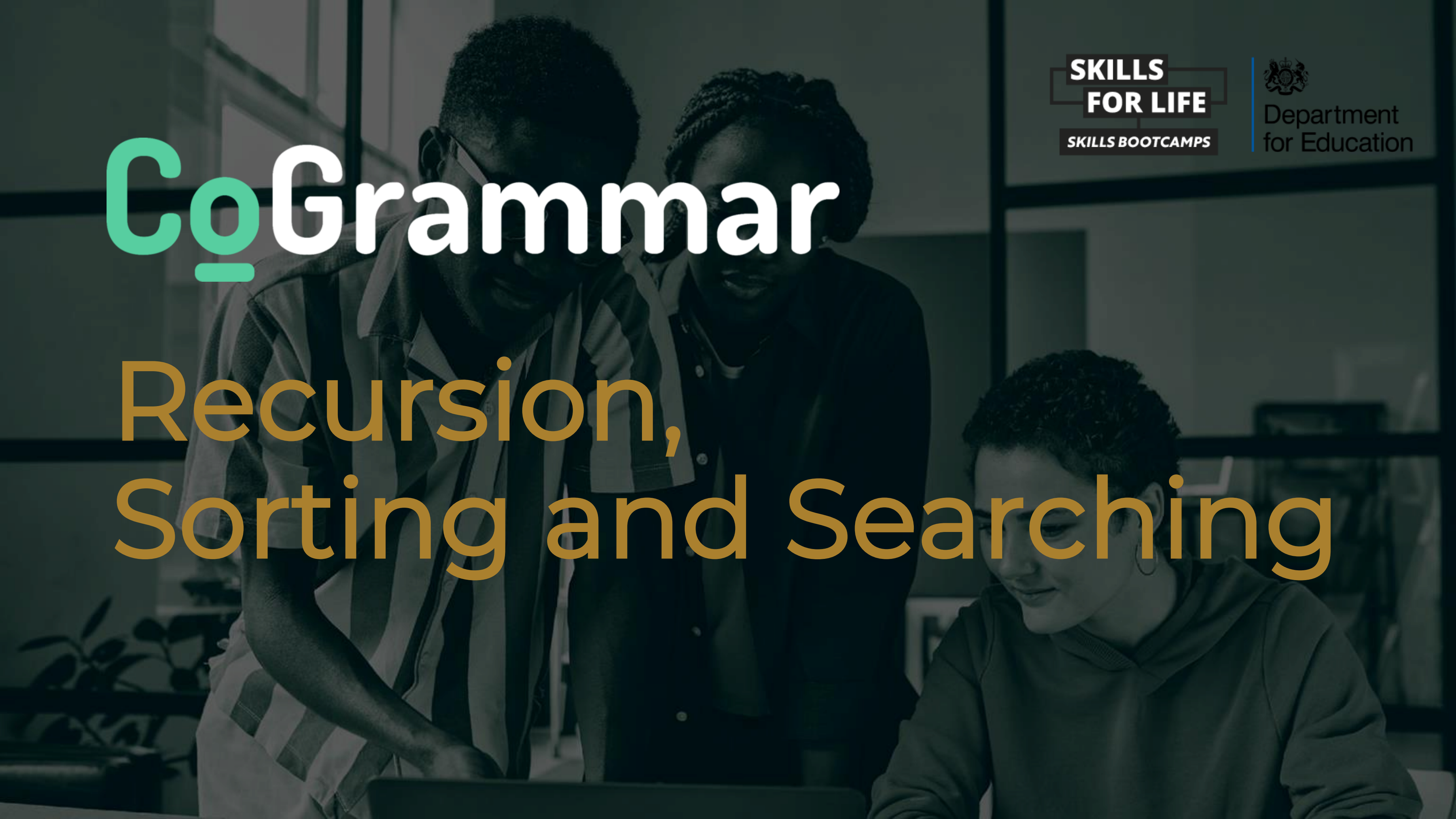
Skills Bootcamp Progression Overview

✓ Criterion 3 - Demonstrating Post-Course Progress

Criterion 3 involves showcasing students' progress after completing the course.
To meet this criterion, students should:

- Complete all mandatory tasks before the bootcamp's end date. This includes any necessary resubmissions, no later than 22 December 2024.
- Achieve at least 84 guided learning hours by the end of the bootcamp, 22 December 2024.

CoGrammar



SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar

Recursion, Sorting and Searching

Learning Outcomes

- Define recursion and identity a recursion problem
- Implement recursion for basic problems like factorial or binary search
- Predict stack overflow from ill-formed recursion
- Describe based sorting algorithms and their associated complexities: Bubble and insertion sort
- Describe basic searching algorithms and their associated complexities: Linear and Binary Search

Recursion

CoGrammar

Recursion Poll

1. **Which of the following best defines a base case in recursive functions?**
   A. The case where the function calls itself
   B. The case that terminates the recursive calls windup
   C. The case where the function returns a value

Recursion Poll

2. **What is the maximum depth of recursion that can be achieved in most programming languages?**

   A. Limited by the size of the call stack

   B. Unlimited, as modern compilers handle recursion efficiently

   C. Limited by the size of the heap memory

CoGrammar

Recursion Poll

3. **When comparing recursive and iterative solutions for the same problem, what are some advantages and disadvantages of each approach?**

   A. Recursion typically uses less memory but may be slower.

   B. Recursion can lead to more elegant and readable code, but iterative solutions are often more efficient in terms of speed and memory usage.

   C. Recursion is always faster and more memory-efficient than iteration.

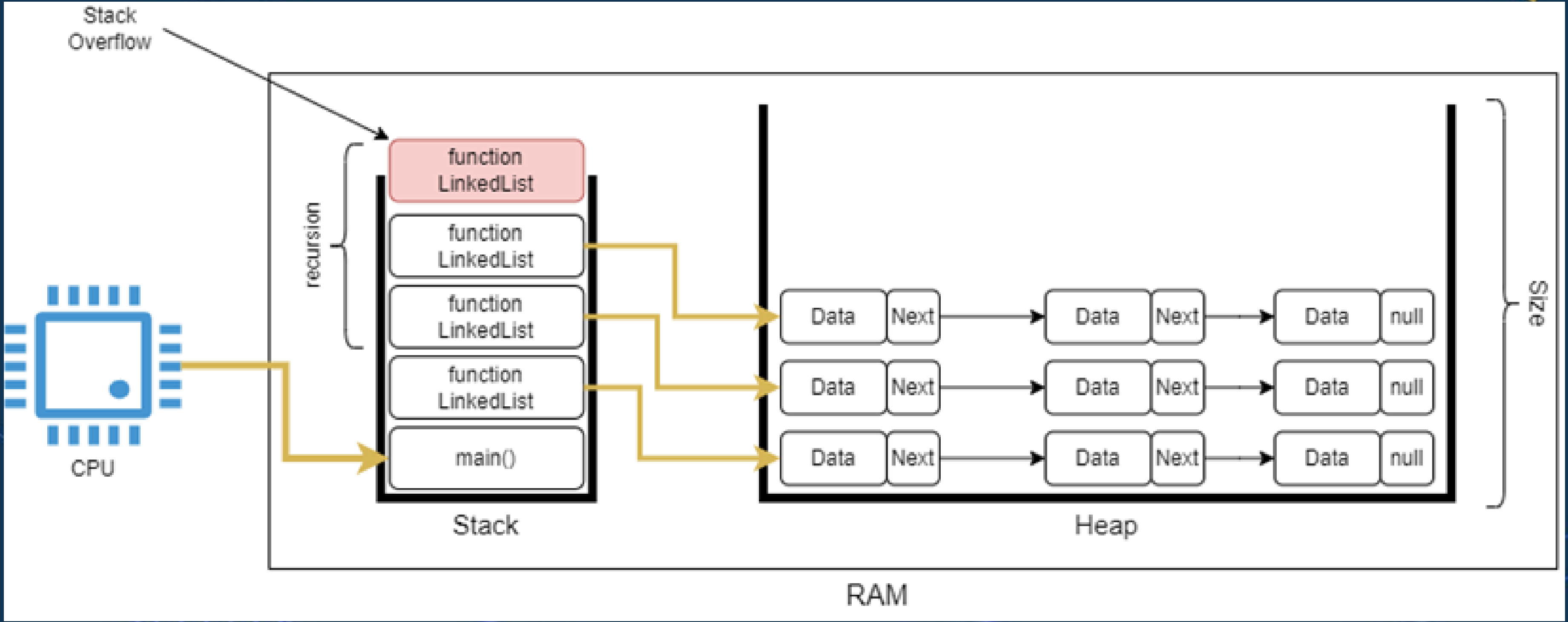
Recursion and Iterations

- Recursion is a programming technique where a function calls itself to solve a problem by breaking it down into smaller, similar sub-problems.
- Iteration is a fundamental programming concept that involves repeating a set of instructions or a process multiple times until a specific condition is met.

Types of iterations

- Count-controlled Iterations
  - Where the number of repetitions is predetermined based on a fixed count or iteration variable.
- Sentinel-controlled Iteration
  - Where the loop continues executing until a specific value known as the "sentinel" is encountered, ie. -1 to exit or EOF.
- Condition-controlled Iterations
  - Where the repetition continues until a specific condition evaluates to false.

Stack Overflow



Let’s get coding!

CoGrammar

Questions and Answers

CoGrammar

Let’s take a short break

Sorting

CoGrammar

Sorting Poll

1. **What is the time complexity of bubble sort?**
   A. O(n^2)
   B. O(n log(n))
   C. O(log(n))

Sorting Poll

2. **What is the main advantage of merge sort over bubble sort?**
   1. Merge sort has a better time complexity (O(n log n))
   2. Merge sort has a smaller memory footprint
   3. Merge sort is easier to implement

Data Structures and Algorithms

- A data structure is a specialised format for organising, processing, retrieving and storing data.

  Eg: **Tree, List, Stacks, Queues**

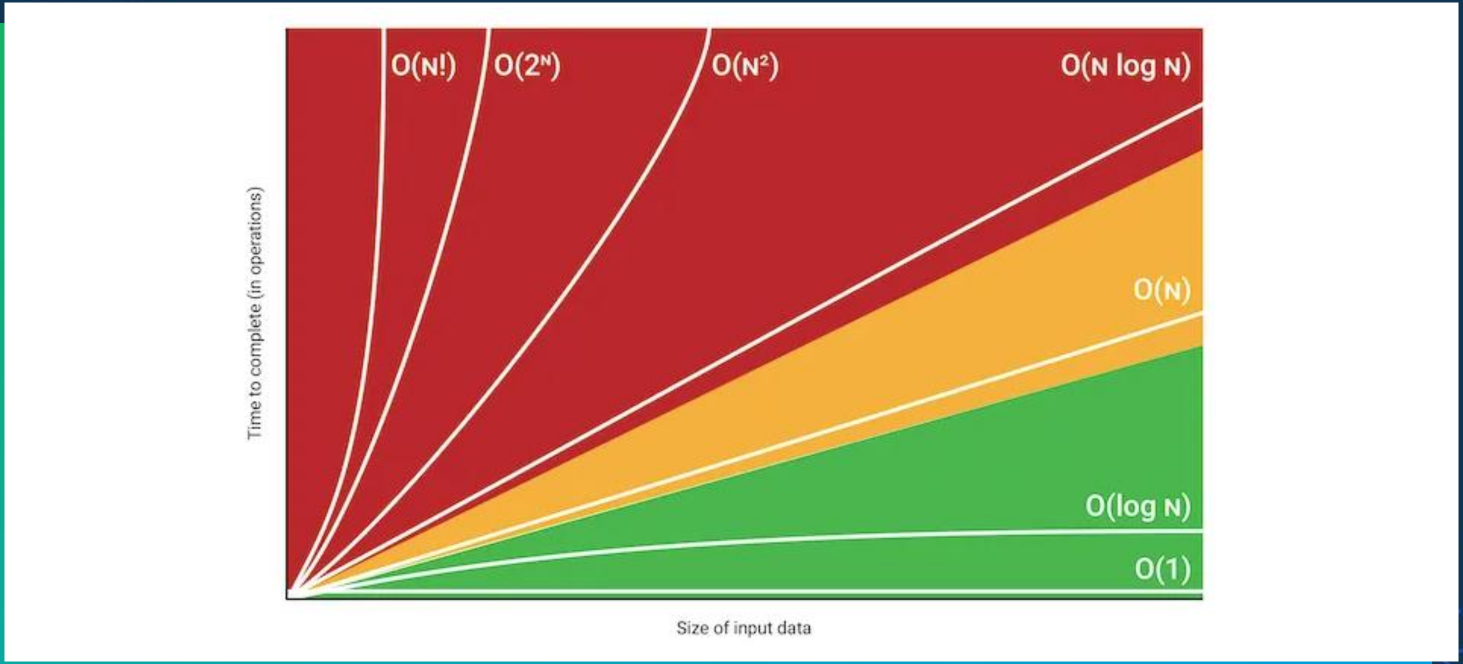
- An algorithm is a set of commands that must be followed for a computer to perform calculations or other problem-solving operations.

  Eg: **Searching, Sorting**

Order of Complexity

- Order of complexity, time complexity or Big-O Notation is the performance or efficiency of an algorithm as the size of its input grows.
- It focuses on the growth rate of the running time or space usage, rather than the exact time, making it possible to compare the efficiency of different algorithms.

Order of Complexity



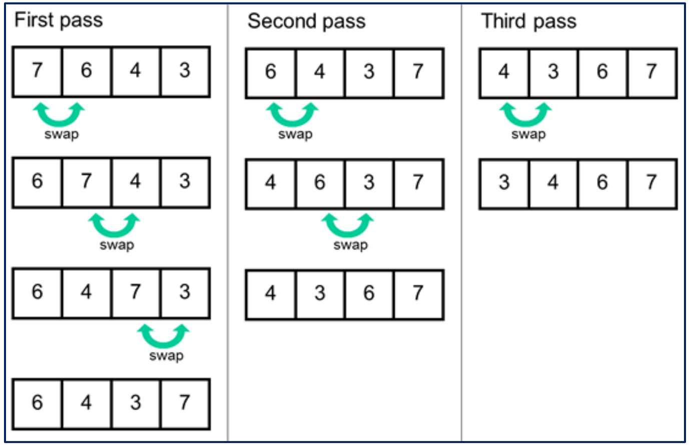
CoGrammar

Sorting Algorithms Definition

- A Sorting Algorithm is used to rearrange a given array or list of elements according to a comparison operator on the elements.

Sorting Algorithms - Bubble

- Bubble sort is a simple sorting algorithm that repeatedly steps through the list, compares adjacent elements, and swaps them if they are in the wrong order, continuing until the list is sorted.



First pass

7 6 4 3

swap

6 7 4 3

swap

6 4 7 3

swap

6 4 3 7

Second pass

6 4 3 7

swap

4 6 3 7

swap

4 3 6 7

Third pass

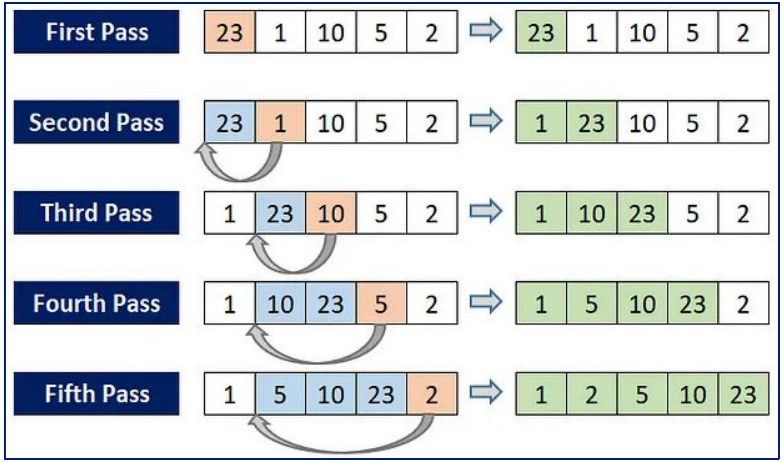
4 3 6 7

swap

3 4 6 7

Sorting Algorithms - Insertion

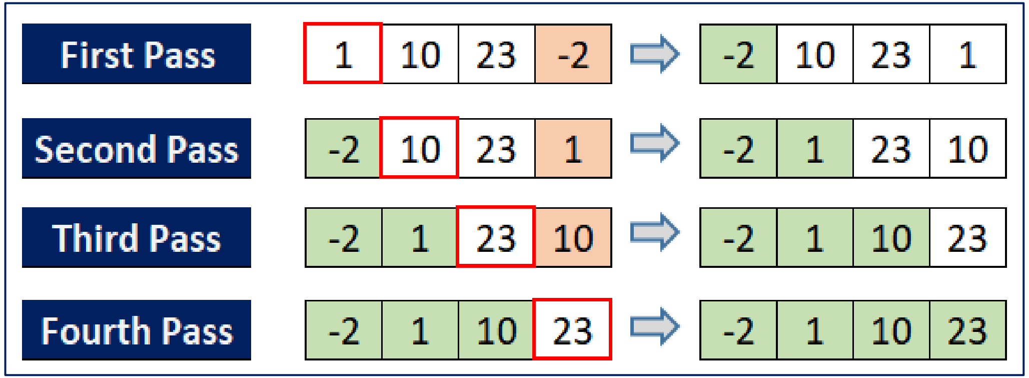
- Insertion sort is a sorting algorithm that builds the final sorted array one item at a time by repeatedly taking the next element and inserting it into the correct position in the already sorted part of the array.



First Pass	23	1	10	5	2	⇨	23	1	10	5	2
Second Pass	23	1	10	5	2	⇨	1	23	10	5	2
Third Pass	1	23	10	5	2	⇨	1	10	23	5	2
Fourth Pass	1	10	23	5	2	⇨	1	5	10	23	2
Fifth Pass	1	5	10	23	2	⇨	1	2	5	10	23

Sorting Algorithms - Selection

- Selection sort is a sorting algorithm that repeatedly selects the minimum element from the unsorted portion of the array and swaps it with the first unsorted element, gradually building up a sorted array from left to right.



First Pass	1	10	23	-2	⇨	-2	10	23	1
Second Pass	-2	10	23	1	⇨	-2	1	23	10
Third Pass	-2	1	23	10	⇨	-2	1	10	23
Fourth Pass	-2	1	10	23	⇨	-2	1	10	23

Let’s get coding!

CoGrammar

Questions and Answers

CoGrammar

Searching

Searching Poll

1. **What does Big O notation do?**
   A. Represents an algorithm's maximum time complexity.
   B. Helps compare how algorithms perform with different input sizes.
   C. Provides an upper bound on worst-case time complexity.

CoGrammar

Searching Poll

2. **What's the main difference between stacks and queues in terms of element access?**

   A. Stacks: Last In, First Out (LIFO); Queues: First In, First Out (FIFO)

   B. Stacks: First In, First Out (FIFO); Queues: Last In, First Out (LIFO)

   C. Both prioritise elements alphabetically.

CoGrammar

Searching Poll

3. **What's the main principle of the binary search algorithm for efficiently finding an element in a sorted array?**

   A. It scans each element of the array linearly until the target is found.

   B. It divides the array into halves, compares with the middle, and narrows down the search space by half until finding the target or exhausting the search.

   C. It sorts the array first and then searches linearly for the target.

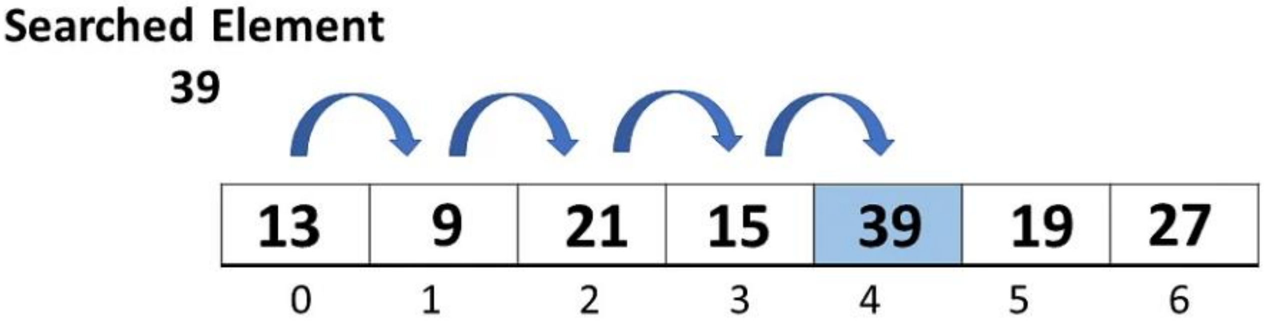
Searching Algorithms Definition

- Searching algorithms are essential tools in computer science used to locate specific items within a collection of data.

CoGrammar

Searching Algorithms - Linear

- Linear search is a simple search algorithm that sequentially checks each element in a list until the target element is found or the end of the list is reached. No sorting is required.



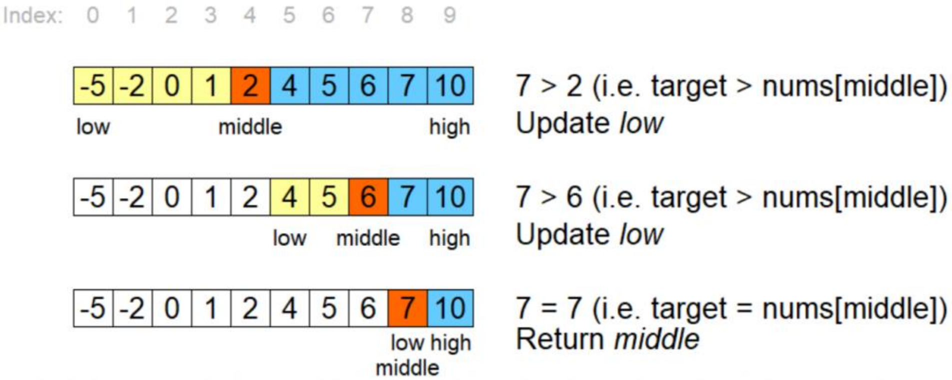
Searched Element

39

13	9	21	15	39	19	27
0	1	2	3	4	5	6

Searching Algorithms - Binary

- Binary search is a search algorithm that efficiently locates a target value within a sorted array by repeatedly dividing the search interval in half and comparing the target value to the middle element, eliminating half of the remaining elements each time.



Index:	0	1	2	3	4	5	6	7	8	9
	-5	-2	0	1	2	4	5	6	7	10
	low				middle					high
	-5	-2	0	1	2	4	5	6	7	10
						low		middle		high
	-5	-2	0	1	2	4	5	6	7	10
									low high middle	

7 > 2 (i.e. target > nums[middle])
Update *low*

7 > 6 (i.e. target > nums[middle])
Update *low*

7 = 7 (i.e. target = nums[middle])
Return *middle*

Let’s get coding!

CoGrammar

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar