CoGrammar

Welcome to this session

Skills Bootcamp:

Tutorial

The session will start shortly...

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

Nurhaan Snyman

Rafiq Manan

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 Full Stack Web Development

- 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 at the end of the session, should you wish to ask any follow-up questions. We will be answering questions as the session progresses as well.
- 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**

CoGrammar

Skills Bootcamp Cloud Web Development

- For all **non-academic questions**, please submit a query: ***www.hyperiondev.com/support***
- **Report a safeguarding incident: *www.hyperiondev.com/safeguardreporting***
- We would love your feedback on lectures: ***Feedback on Lectures.***
- Find all the lecture **content** in your **Lecture Backpack** on GitHub.
- If you are hearing impaired, kindly use your computer's function through Google chrome to enable captions.

Skills Bootcamp Progression Overview

✅ Criterion 1 - Initial Requirements

Specific achievements within the first two weeks of the program.

To meet this criterion, students need to, by no later than **01 December 2024 (C11)** or **22 December 2024 (C12):**

- **Guided Learning Hours** (GLH): Attend a minimum of 7-8 GLH per week (lectures, workshops, or mentor calls) for a total minimum of **15 GLH**.
- **Task Completion:** Successfully complete the **first 4 of the assigned tasks**.

✅ Criterion 2 - Mid-Course Progress

Progress through the successful completion of tasks within the first half of the program.

To meet this criterion, students should, by no later than **12 January 2025 (C11)** or **02 February 2025 (C12):**

- **Guided Learning Hours** (GLH): Complete at least **60 GLH**.
- **Task Completion :** Successfully complete the **first 13 of the assigned tasks**.

CoGrammar

Skills Bootcamp Progression Overview

✅ Criterion 3 – End-Course Progress

Showcasing students' progress nearing the completion of the course.

To meet this criterion, students should:

- **Guided Learning Hours** (GLH)**:** Complete the **total minimum required GLH,** by the **support end date.**
- **Task Completion : Complete all mandatory tasks**, including any necessary resubmissions, by the end of the bootcamp, **09 March 2025 (C11)** or **30 March 2025 (C12).**

✅ Criterion 4 - Employability

Demonstrating progress to find employment.

To meet this criterion, students should:

- **Record an Interview Invite:** Students are required to record proof of invitation to an interview by **30 March 2025 (C11)** or **04 May 2025 (C12)**.
  - **South Holland Students** are required to proof and interview by **17 March 2025.**
- **Record a Final Job Outcome :** Within 12 weeks post-graduation, students are required to record a job outcome.

CoGrammar

Stay Safe Series:

Mastering Online Safety One week at a Time

While the digital world can be a wonderful place to make education and learning accessible to all, it is unfortunately also a space where harmful threats like online radicalization, extremist propaganda, phishing scams, online blackmail and hackers can flourish.

As a component of this BootCamp the *Stay Safe Series* will guide you through essential measures in order to protect yourself & your community from online dangers, whether they target your privacy, personal information or even attempt to manipulate your beliefs.

Don’t Take the Bait:
How to Spot Phishing Scams

- Check the Sender's Email Address
- Look for Generic Greetings
- Be Wary of Urgent Language
- Hover Over Links
- Inspect Attachments Carefully
- Look for Spelling and Grammar Errors
- Verify with the Source
- Use Multi-Factor Authentication
- Stay Informed
- Report Suspicious Emails

What is an Algorithm?

A. A detailed step-by-step procedure for solving a problem
B. A programming language
C. A hardware component
D. A design pattern

Which of the following is a characteristic of a good algorithm?

A. Complexity and Length
B. Correctness, Efficiency, and Readability
C. Use of Multiple Programming Languages
D. Randomness in Output

CoGrammar

Learning Outcomes

- Explain the concept of sorting and searching algorithms.
- Write and explain basic algorithms like Bubble Sort, Selection Sort, and Binary Search.
- Optimise and test basic algorithms with real-world coding problems.

Lecture Overview

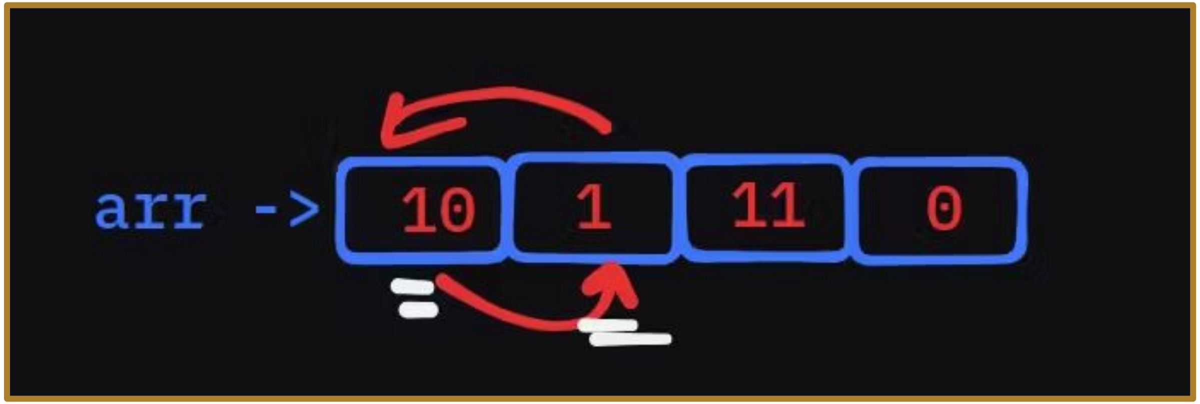
- → Sorting Algorithms
- → Break
- → Searching Algorithms
- → Assessment and Q&A

Sorting Algorithms

- Sorting helps in organizing data for better searching and analysis.
- Focus:
  - Bubble Sort
  - Selection Sort
  - Merge Sort.

Bubble Sort

- Repeatedly steps through the list, compares adjacent pairs, and swaps if they are out of order.

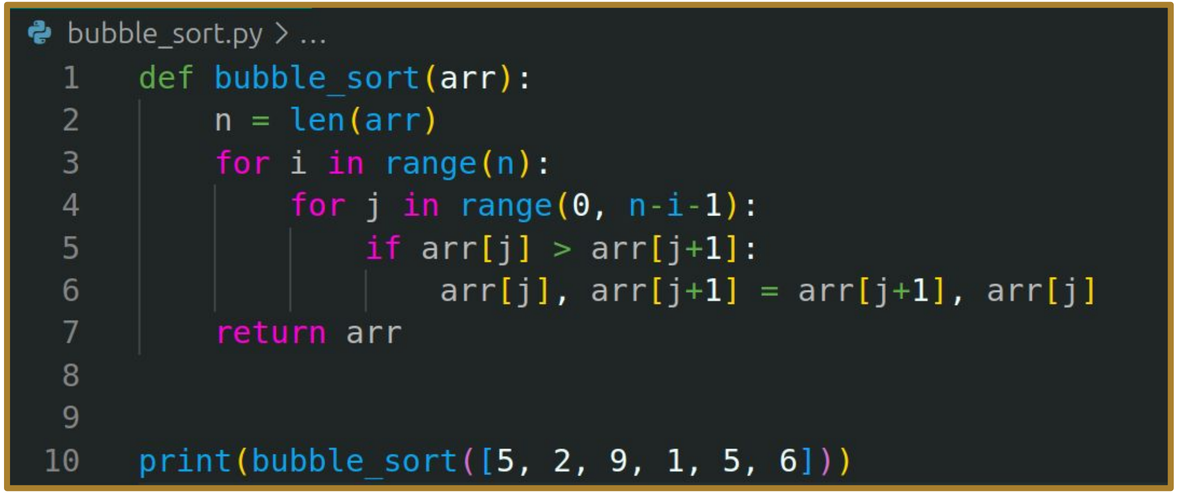


Bubble Sort

- Let's visualize the Bubble Sort process using a simple example array: [64, 34, 25, 12, 22, 11, 90].
- Step-by-Step Example
- Pass 1:
  - Compare 64 and 34 → Swap: [34, 64, 25, 12, 22, 11, 90]
  - Compare 64 and 25 → Swap: [34, 25, 64, 12, 22, 11, 90]
  - Compare 64 and 12 → Swap: [34, 25, 12, 64, 22, 11, 90]
  - Compare 64 and 22 → Swap: [34, 25, 12, 22, 64, 11, 90]
  - Compare 64 and 11 → Swap: [34, 25, 12, 22, 11, 64, 90]
  - Compare 64 and 90 → No swap: [34, 25, 12, 22, 11, 64, 90]
- Pass 2:
  - Repeat the process until the entire array is sorted.

Bubble Sort

- Repeatedly steps through the list, compares adjacent pairs, and swaps if they are out of order.



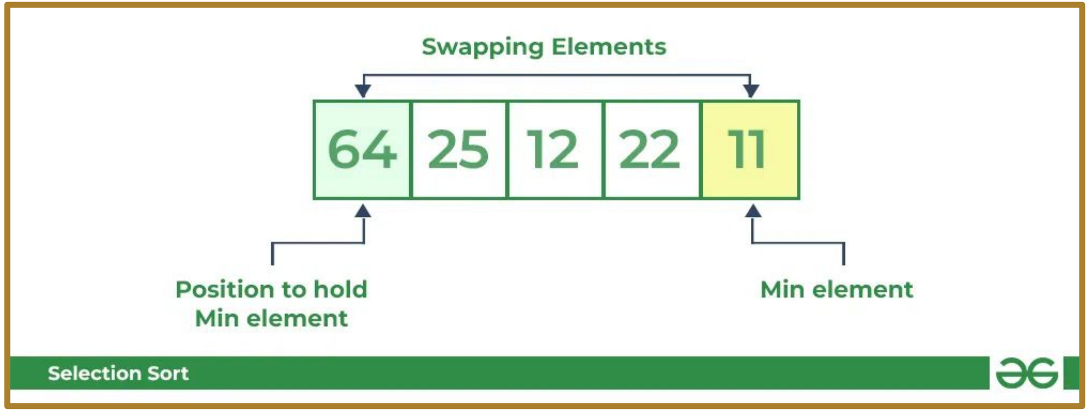
bubble_sort.py > ...

```python
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]
    return arr


print(bubble_sort([5, 2, 9, 1, 5, 6]))
```

Selection Sort

- Select the smallest element from the unsorted part and swap it with the first unsorted



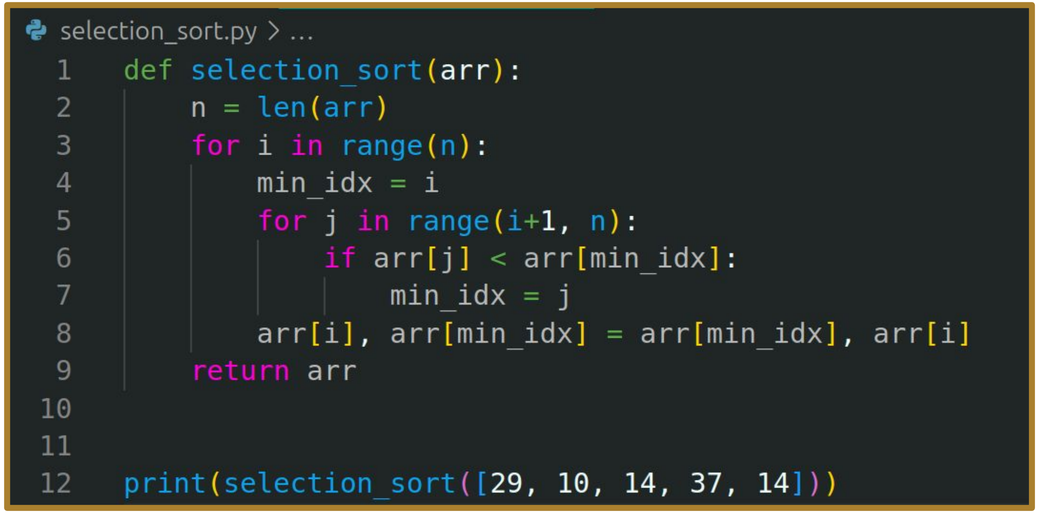
CoGrammar

Selection Sort

- Array: [64, 25, 12, 22, 11]
- Pass 1:
  - Find the smallest element (11) and swap it with the first element (64).
  - Array after swap: [11, 25, 12, 22, 64]
- Pass 2:
  - Find the smallest element (12) in the unsorted section [25, 12, 22, 64] and swap it with 25.
  - Array after swap: [11, 12, 25, 22, 64]
- Pass 3:
  - Find the smallest element (22) in [25, 22, 64] and swap it with 25.
  - Array after swap: [11, 12, 22, 25, 64]
- Pass 4:
  - Swap the last two elements if needed. No change here, as 25 < 64.
  - Final Sorted Array: [11, 12, 22, 25, 64]

Selection Sort

- Select the smallest element from the unsorted part and swap it with the first unsorted

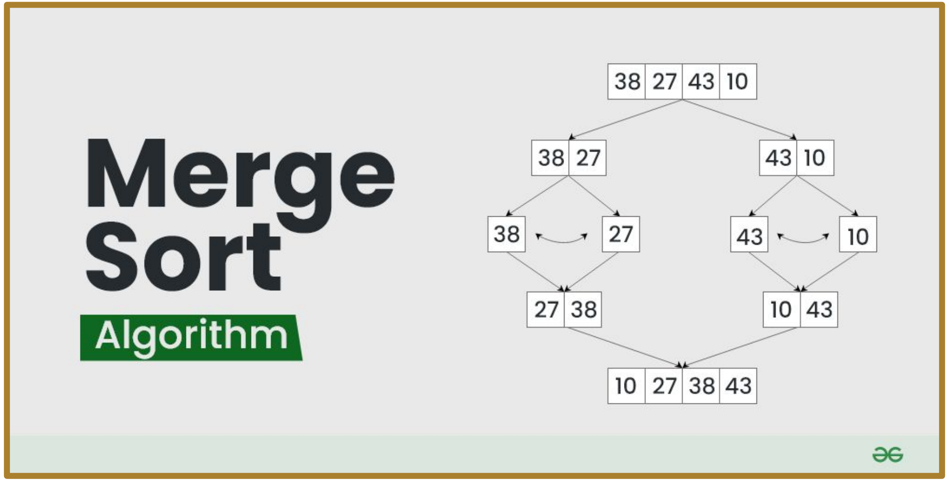


```python
def selection_sort(arr):
    n = len(arr)
    for i in range(n):
        min_idx = i
        for j in range(i+1, n):
            if arr[j] < arr[min_idx]:
                min_idx = j
        arr[i], arr[min_idx] = arr[min_idx], arr[i]
    return arr


print(selection_sort([29, 10, 14, 37, 14]))
```

Merge Sort

- Divide the list into halves, sort each half, and merge them back together.



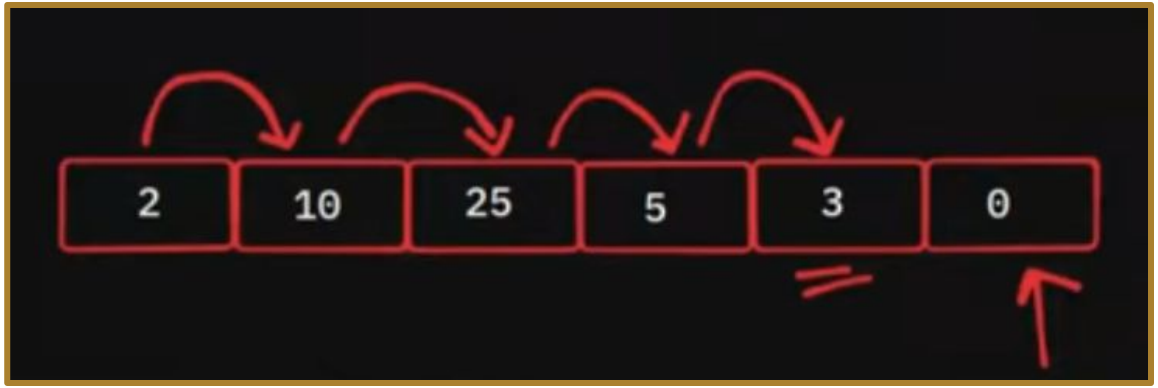
Searching Algorithms

- Searching helps find elements in a dataset.
- Focus:
  - Linear Search
  - Binary Search.

Let's take a break

Linear Search

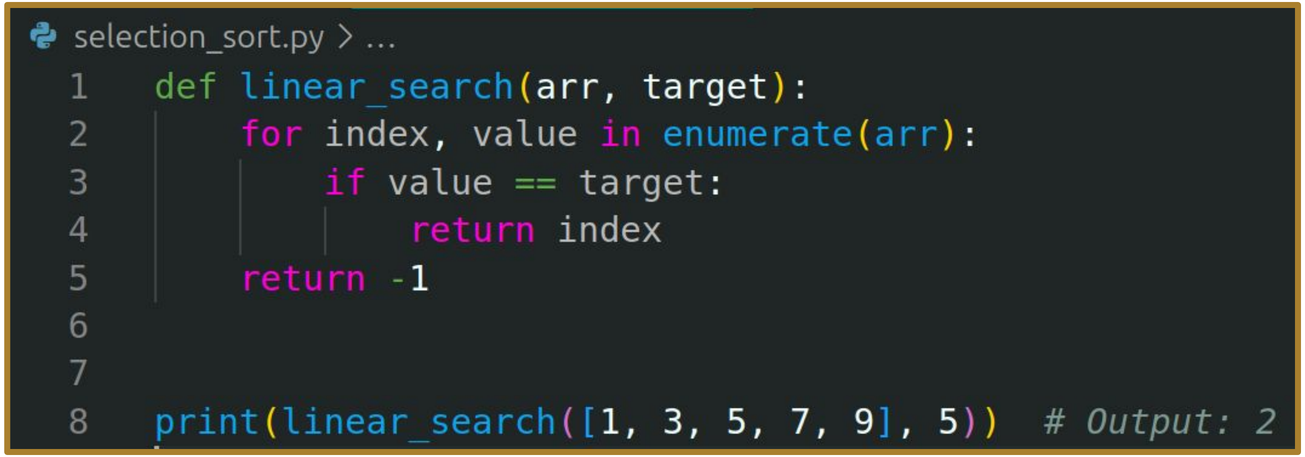
- Check every element until the target is found.



CoGrammar

Linear Search

❖ Check every element until the target is found.

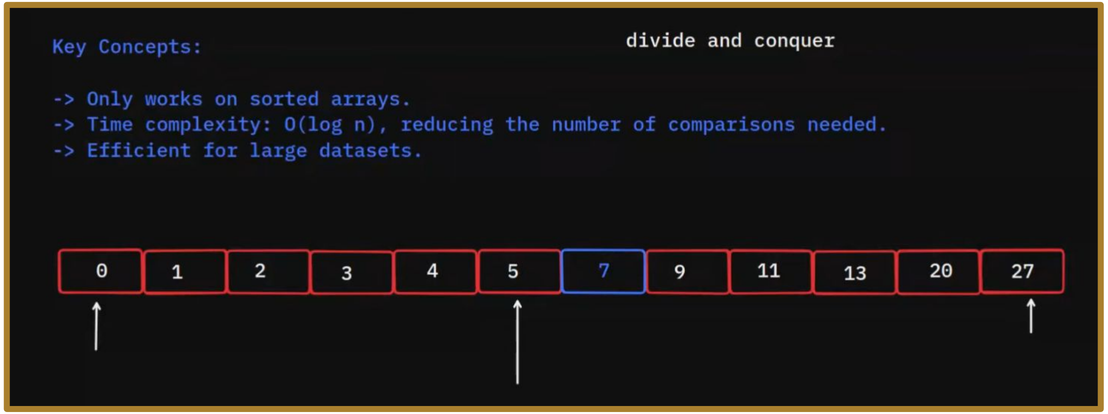


```python
def linear_search(arr, target):
    for index, value in enumerate(arr):
        if value == target:
            return index
    return -1


print(linear_search([1, 3, 5, 7, 9], 5))  # Output: 2
```

Binary Search

- ❖ Divide the list into halves and eliminate half each iteration.
- ❖ **Note**:
  - ➢ Requires a sorted list.



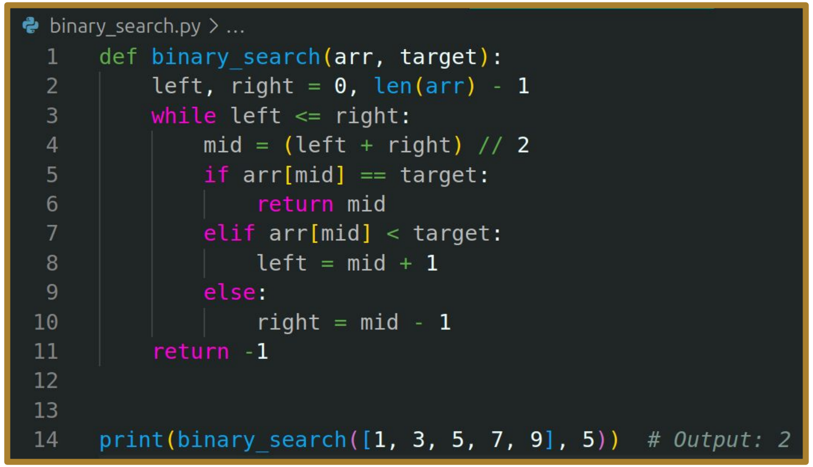
```
Key Concepts:                                    divide and conquer

-> Only works on sorted arrays.
-> Time complexity: O(log n), reducing the number of comparisons needed.
-> Efficient for large datasets.
```

0	1	2	3	4	5	7	9	11	13	20	27

Binary Search

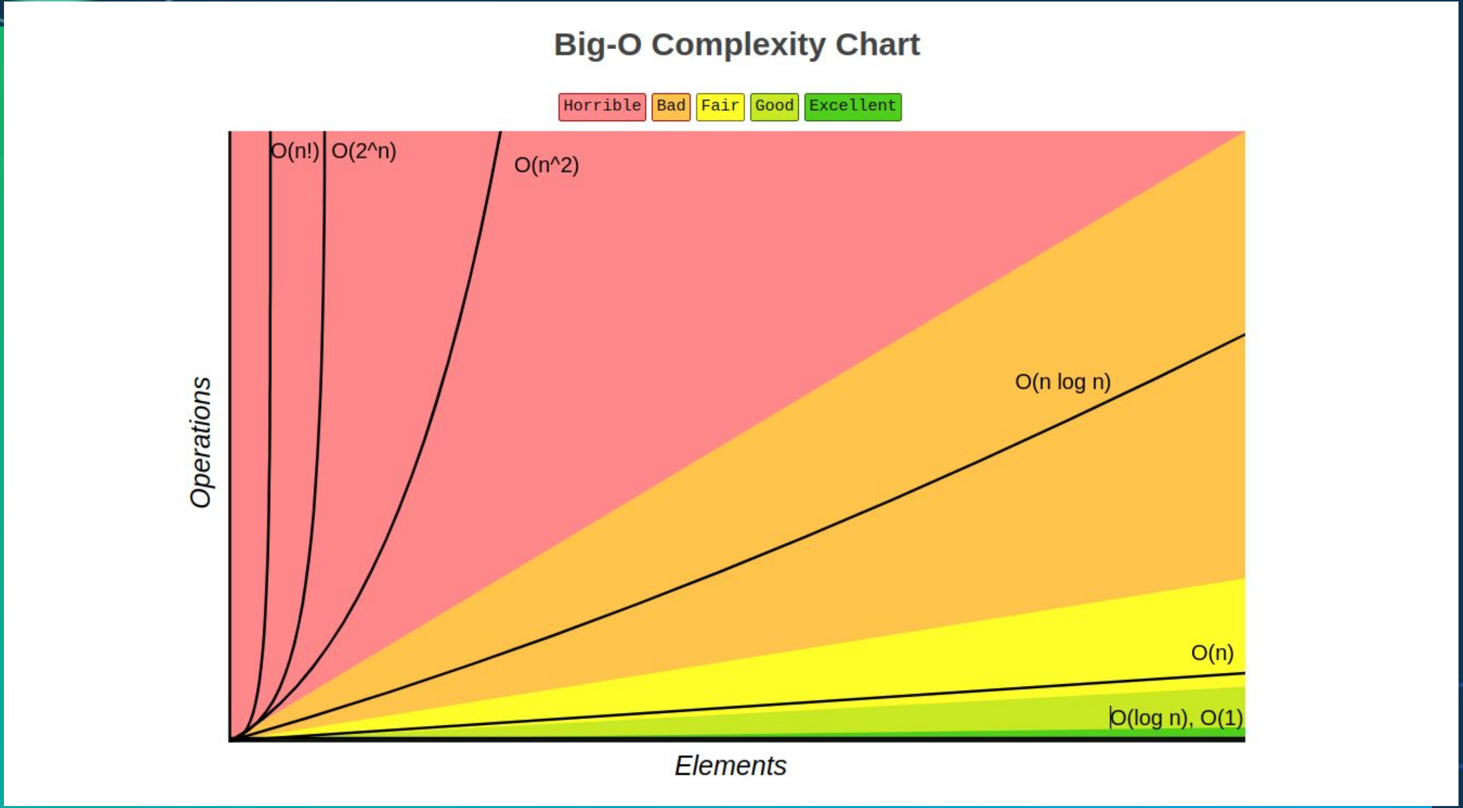
- Divide the list into halves and eliminate half each iteration.
- **Note**:
  - Requires a sorted list.



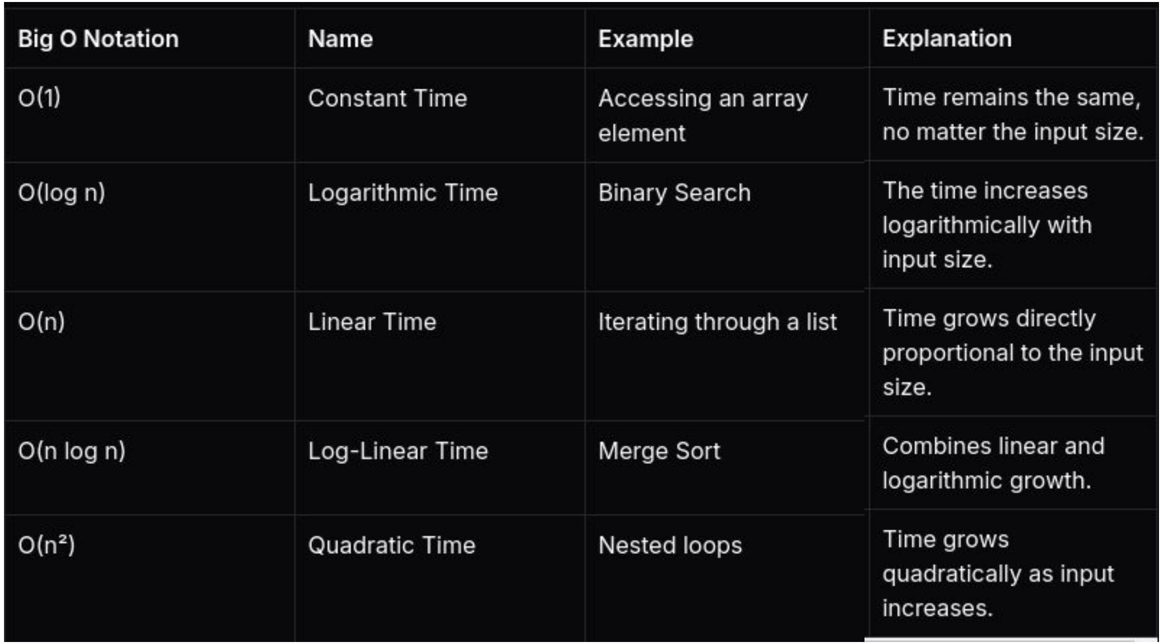
```python
def binary_search(arr, target):
    left, right = 0, len(arr) - 1
    while left <= right:
        mid = (left + right) // 2
        if arr[mid] == target:
            return mid
        elif arr[mid] < target:
            left = mid + 1
        else:
            right = mid - 1
    return -1


print(binary_search([1, 3, 5, 7, 9], 5))  # Output: 2
```

Time and Space Complexity



Common Big O Notations



Big O Notation	Name	Example	Explanation
O(1)	Constant Time	Accessing an array element	Time remains the same, no matter the input size.
O(log n)	Logarithmic Time	Binary Search	The time increases logarithmically with input size.
O(n)	Linear Time	Iterating through a list	Time grows directly proportional to the input size.
O(n log n)	Log-Linear Time	Merge Sort	Combines linear and logarithmic growth.
$O(n^2)$	Quadratic Time	Nested loops	Time grows quadratically as input increases.

Review and Next Steps

- Key Takeaways:
  - Sorting: Bubble Sort, Selection Sort, Merge Sort.
  - Searching: Linear Search, Binary Search.
  - Time and Space Complexity Overview.
- Next Steps:
  - Continue practicing
  - Explore real-world applications

Additional Resources

- Python Documentation:
  - https://python.org
- Algorithm Visualizer:
  - https://visualgo.net
- Practice Problems:
  - https://leetcode.com

What is the primary difference between Linear Search and Binary Search?

A. Linear Search only works on sorted lists, while Binary Search works on unsorted lists.

B. Binary Search repeatedly halves the list, while Linear Search checks elements one by one.

C. Linear Search is faster than Binary Search.

D. Binary Search compares each element individually.

After performing one complete pass of Bubble Sort on the list [5, 2, 8, 6, 1], what will the list look like?

A. [2, 5, 8, 6, 1]
B. [2, 5, 6, 1, 8]
C. [5, 2, 6, 8, 1]
D. [2, 8, 6, 1, 5]

CoGrammar

Questions and Answers

CoGrammar

Thank you for attending

CoGrammar

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education