

# Sorting & Big-O notation

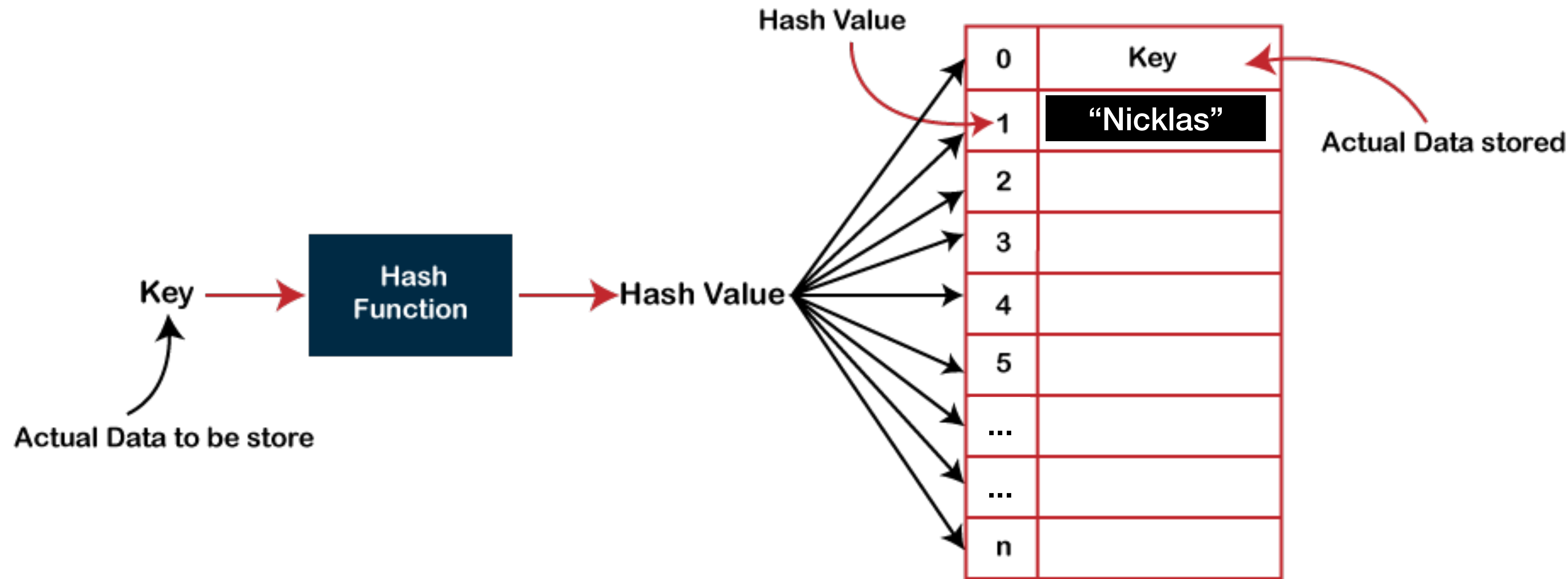
Applikationsudvikling - CS101

# Agenda

## Sorting & Big-O notation

- Time complexity Big-O notation
- Trading space for time “leetcode” for twosum
- Sorting algorithms
  - Bubble sort algorithm Pair Analysis
  - Insertion sort algorithm Pair programming
  - Selection sort algorithm Pair programming

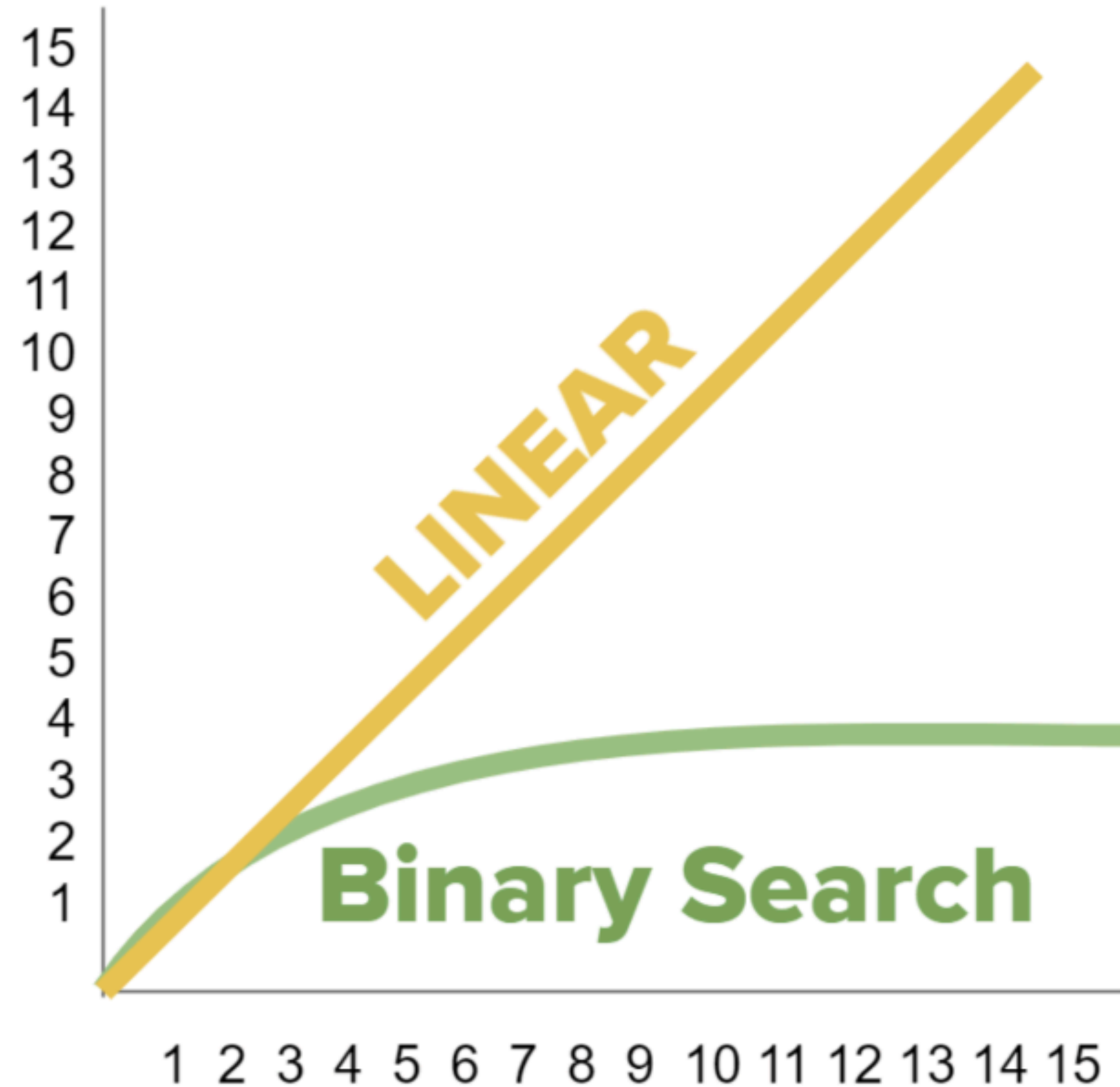
Hash Value	Key	Value
1	Nicklas	20436262
3	Karsten	20202020
12	Evander	29392291
	...	...
	N	N



What is meant by fast?

# Order of growth

Time complexity



# Order of growth

## TwoSum function

```
fun twoSum(nums: IntArray, target: Int): IntArray {  
    for (i in nums.indices) {  
        for (j in i + 1 until nums.size) {  
            if (nums[i] + nums[j] == target) {  
                return intArrayOf(i, j)  
            }  
        }  
    }  
    throw IllegalArgumentException("No two sum solution")  
}
```

# Order of growth

TwoSum function

Target: 4

1

0

7

2

5

2

5



# Order of growth

TwoSum function



Target: 4



Not only 3 increase but  $3 \times 3 = 3^2 - 2$

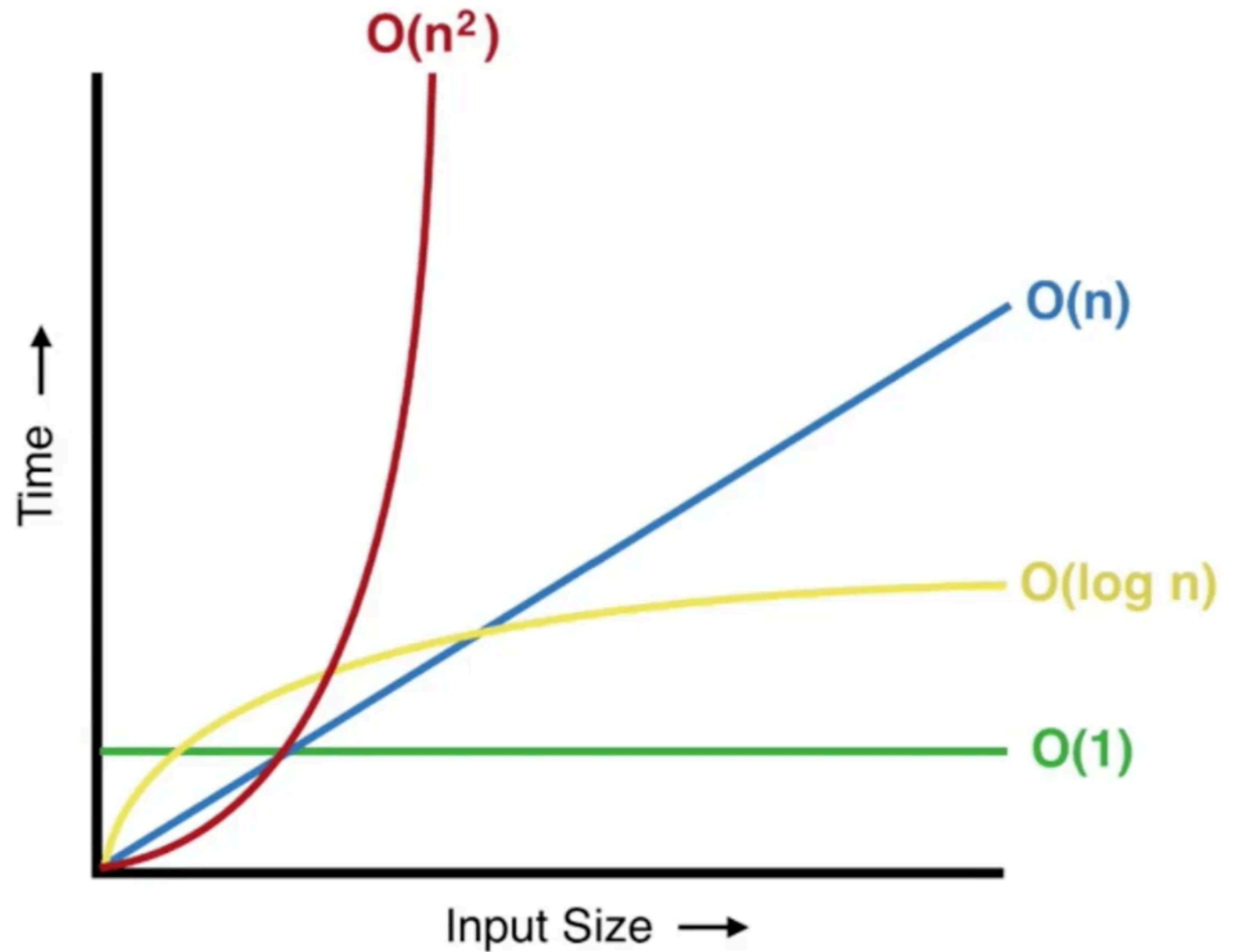


1 0 7 0 5 0 6 2 2

Not only 3 increase but  $3*3 = 3^2 - 2$

Solution has order of  
growth  
 $O(n^2)$

## Big O Notation



# How does **input** influence execution time?

TwoSum function - observe the loops

$O(n^2)$

```
fun twoSum(nums: IntArray, target: Int): IntArray {  
    for (i in nums.indices) {  
        for (j in i + 1 until nums.size) {  
            if (nums[i] + nums[j] == target) {  
                return intArrayOf(i, j)  
            }  
        }  
    }  
    throw IllegalArgumentException("No two sum solution")  
}
```

Algorithm solution:  
Solving twosum in a single iteration

## Sequential access

Diagram illustrating a sequence of numbers: 1, 0, 7, 2, 5, 2. Arches connect the first four numbers (1, 0, 7, 2). An orange box is positioned above the number 2.

Diagram illustrating a hash table structure using separate chaining:

- Key**: Input data to be stored.
- Hash Function**: Processes the key to generate a **Hash Value**.
- Hash Value**: Points to a specific slot in the hash table.
- Hash Table**: A table with slots indexed from 0 to n. Each slot contains a **Key** and a pointer to a linked list of **Actual Data stored**.
- Random access**: A yellow box indicating the type of access supported by this structure.

N = 9

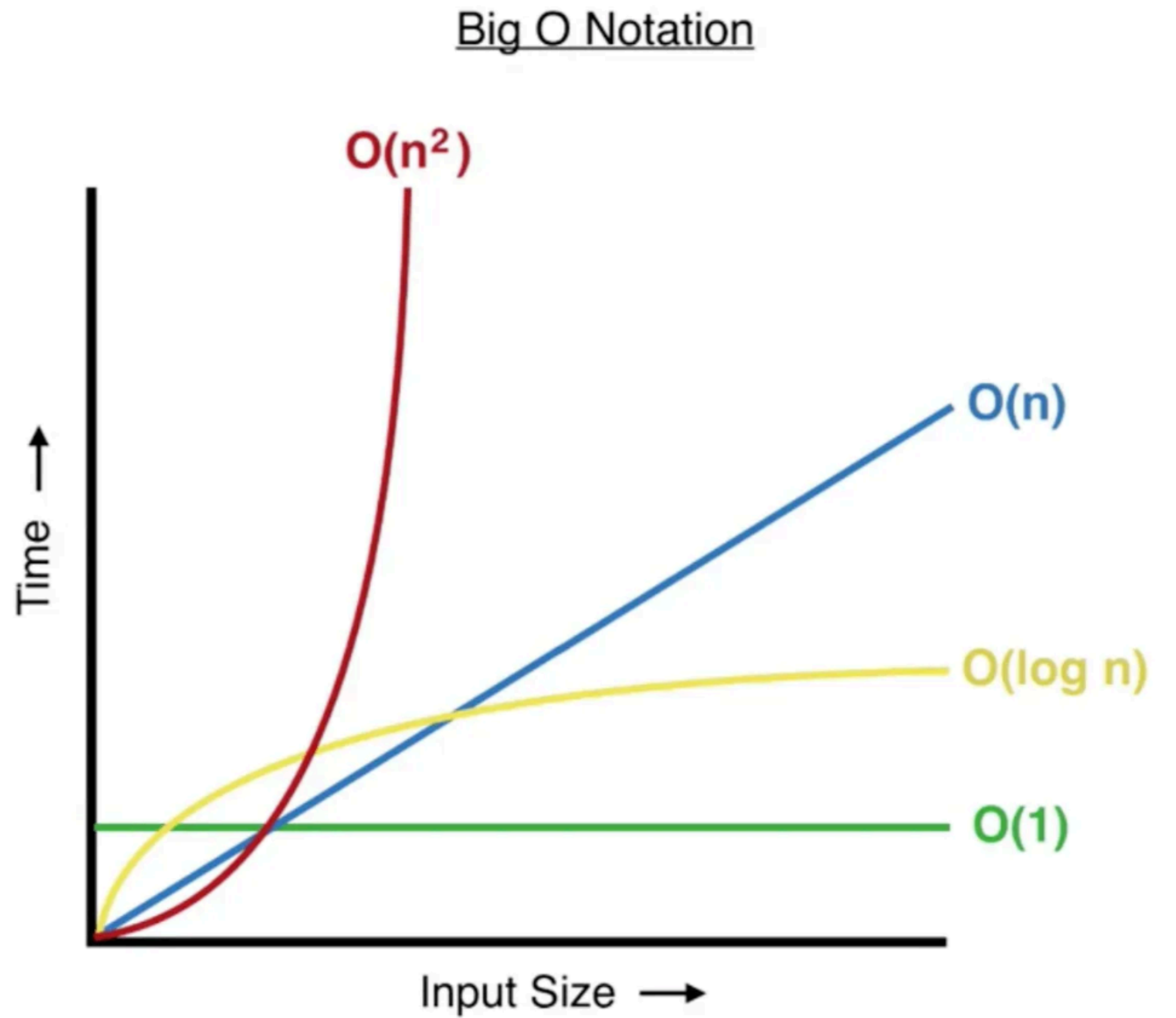
1	0	7	0	5	0	6	2	2
---	---	---	---	---	---	---	---	---

N = 6

1	0	7	0	5	0			
---	---	---	---	---	---	--	--	--

Runtime =  $O(6 \cdot 9)$

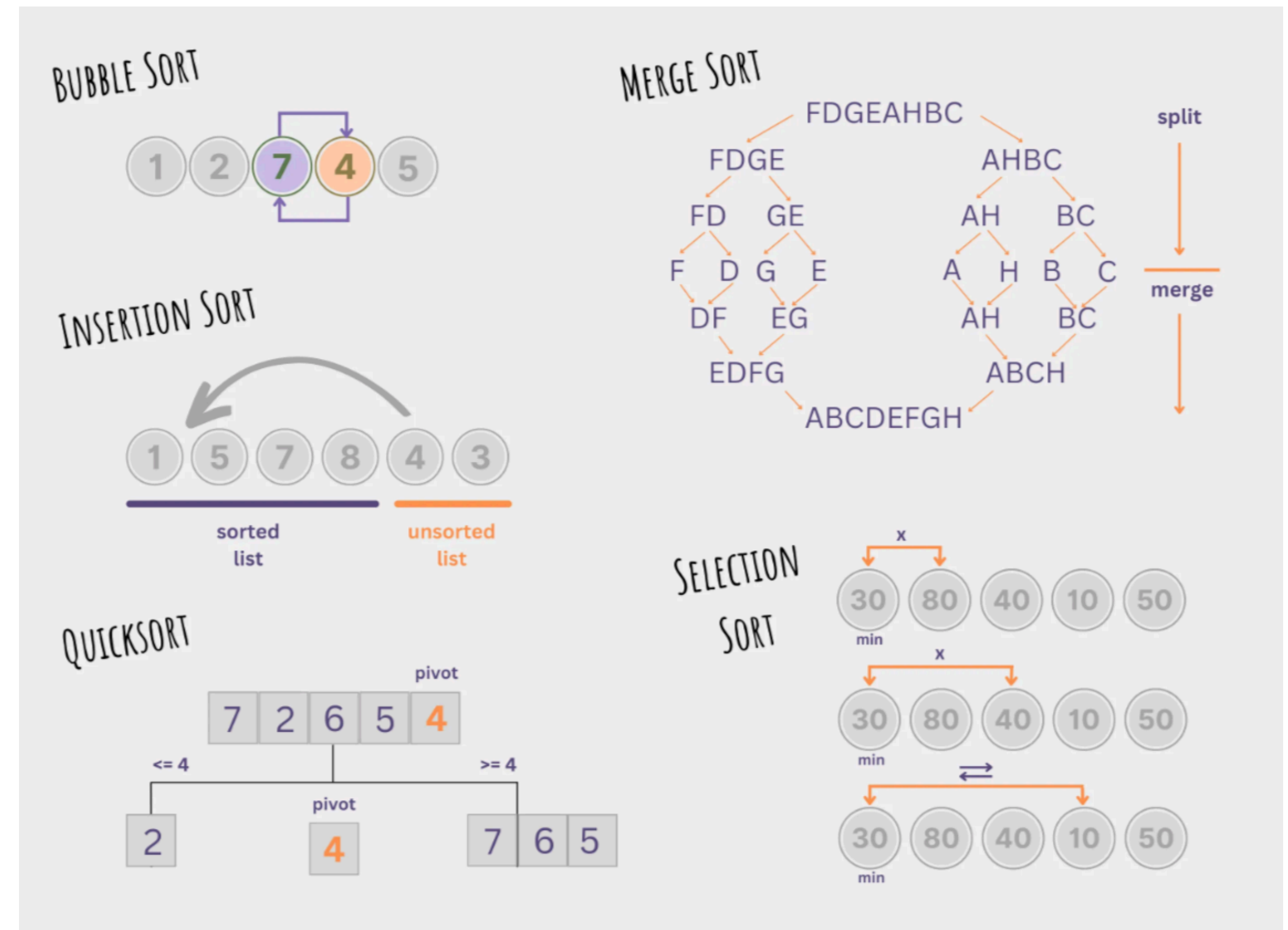
# Big-O exercises



# Sorting algorithms

## CS101

- We will be working with 3 simple sorting algorithms
  - Bubble sort
  - Selection sort
  - Insertion sort
- You will be implementing **selection sort & insertion sort** in a pair programming fashion





# Bubble-sort example