TODO BEFORE

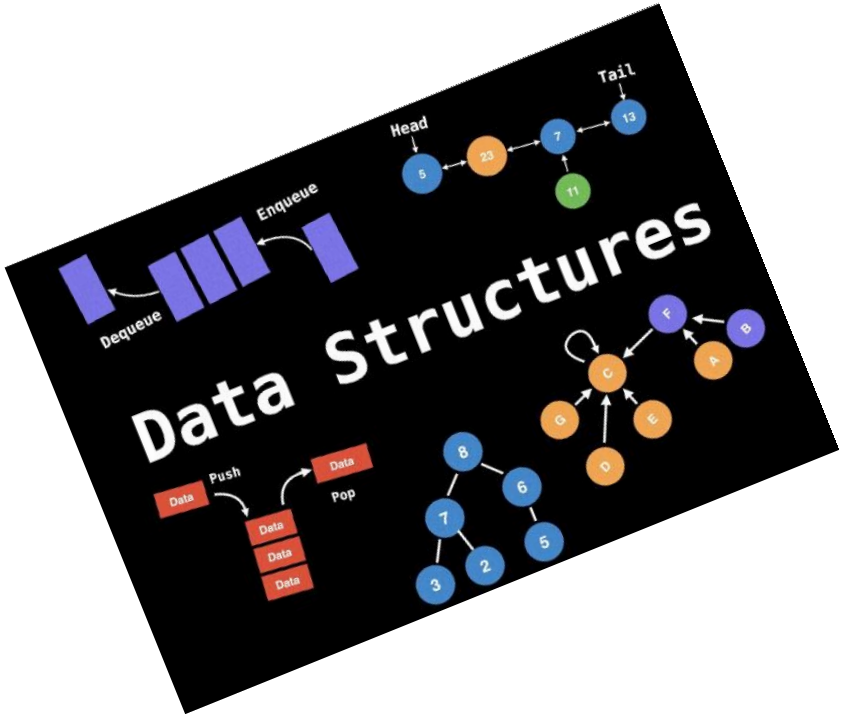
Useful Resource #BeReady

BEFORE NEXT SESSION

- ✓ Watch this first video about ADTs
- ✓ Also watch this video

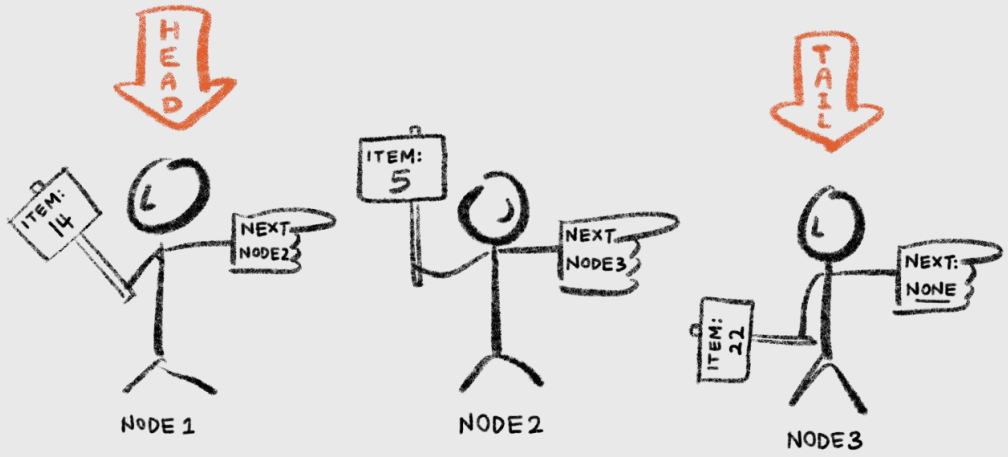
ALONG THE COURSE !

- ✓ Follow this playlist to understand the most important data structures



ADVANCED ALGORITHM

W4-S2 – Linked List



CADT IDT

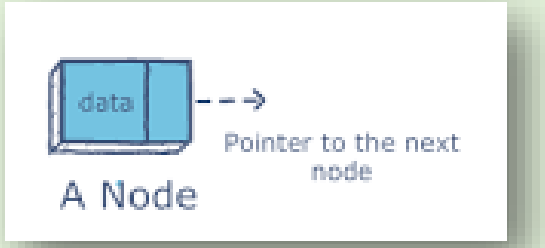
Objectives for today

- Understand the Concept of **Linked List**
- Differentiate Between **Array** and **Linked List**
- Define the operations of a **Linked List** and their **complexity**

EXPLAIN

Linked List

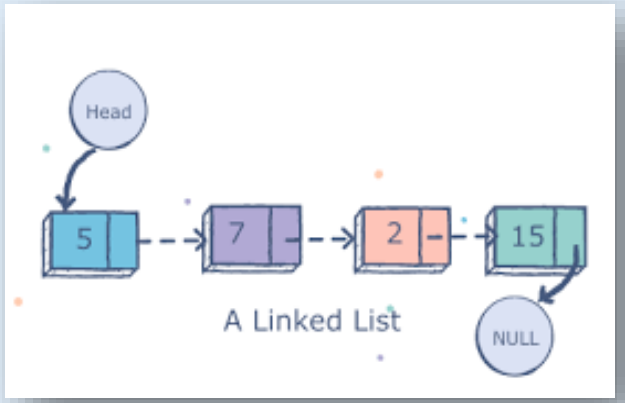
A data structure that can **store** indefinite amount of elements. Each **element** is **linked** with **one another** via a **link** (**pointer**).



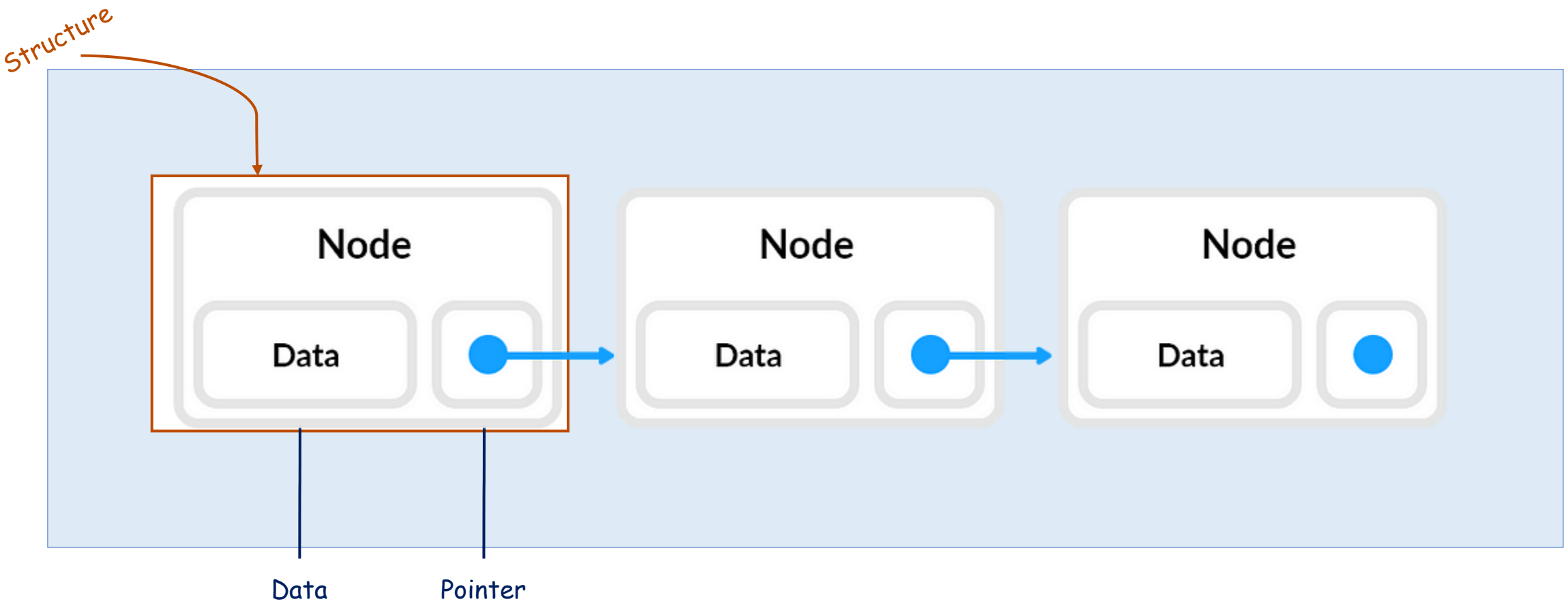
EXPLAIN

Linked List (LL)

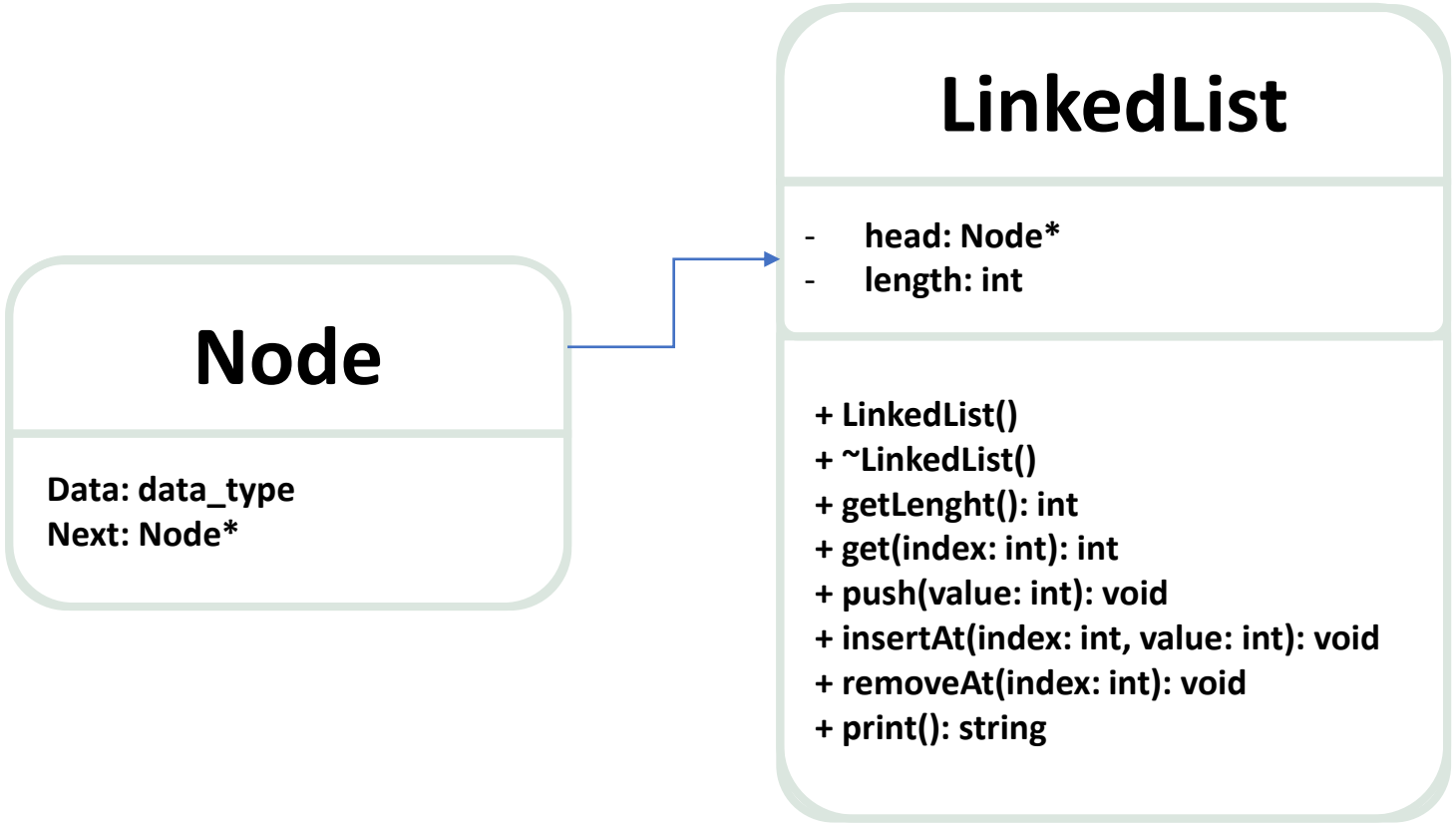
- A sequence of data structures, which are connected together via links.
- A sequence of links which contains items. Each link contains a connection to another link.



Singly Linked List (SLL)



Singly Linked List (SLL)



Array vs Linked List

Array	Linked list
• Limited size (static)	• Unlimited size (dynamic)
• Data can be accessed randomly	• Data can be accessed from head or tail
• Allocate memory at compiled time	• Memory utilization is efficient
• Inefficient Memory utilization	• Efficient Memory utilization (A new element is created dynamically at run time)

Operations with Linked list

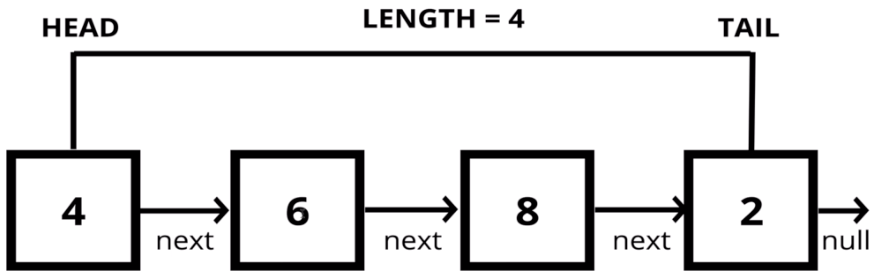
Let s describe **the specifications** of 3 operations on this ADT

OPERATIONS

1. int **get**(index)
2. void **insertAt**(value)
3. void **removeAt**(index)

Linked List

Singly Linked Lists



GAME

X 05 50min

THE LINKED LIST SHOW

- ✓ Group of **7 students**
- ✓ Each group is responsible of 1 operation of linked list (*Insert, search, delete*)

1 - PREPARATION

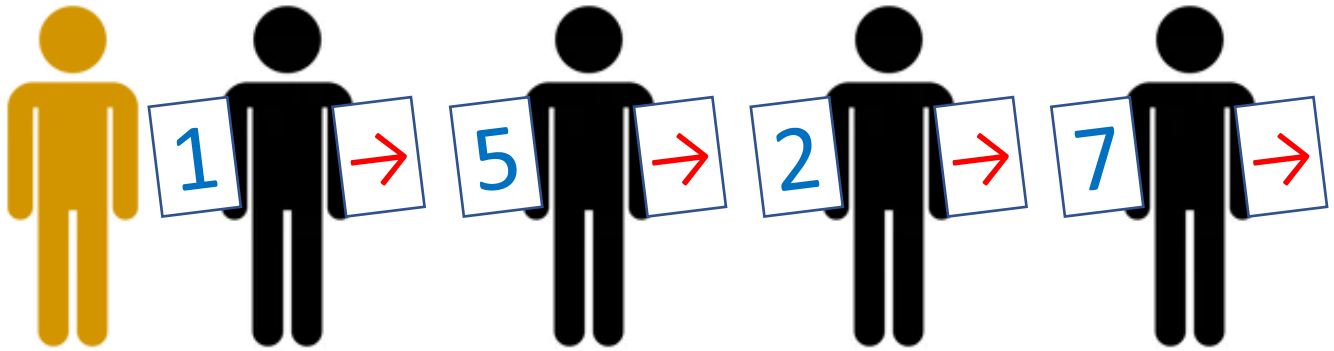
- ✓ **40 min** : Learn the Linked List Operation
  - ✓ Use resources: internet, videos, etc.
  - ✓ Goal: Everyone in the group should understand their assigned operation.
- ✓ **10 min** : Prepare the show !

2 - SHOW

- ✓ Every group present their show
  - ✓ Stand in a line and demonstrate your linked list operation (Insert, Search, or Delete).
  - ✓ Props: Arrange yourselves and use arrows to show pointer connections.
- ✓ A **speaker** can explain the steps
  - ✓ A designated speaker explains each step of the operation.
  - ✓ Show how pointers are adjusted as the operation is performed.

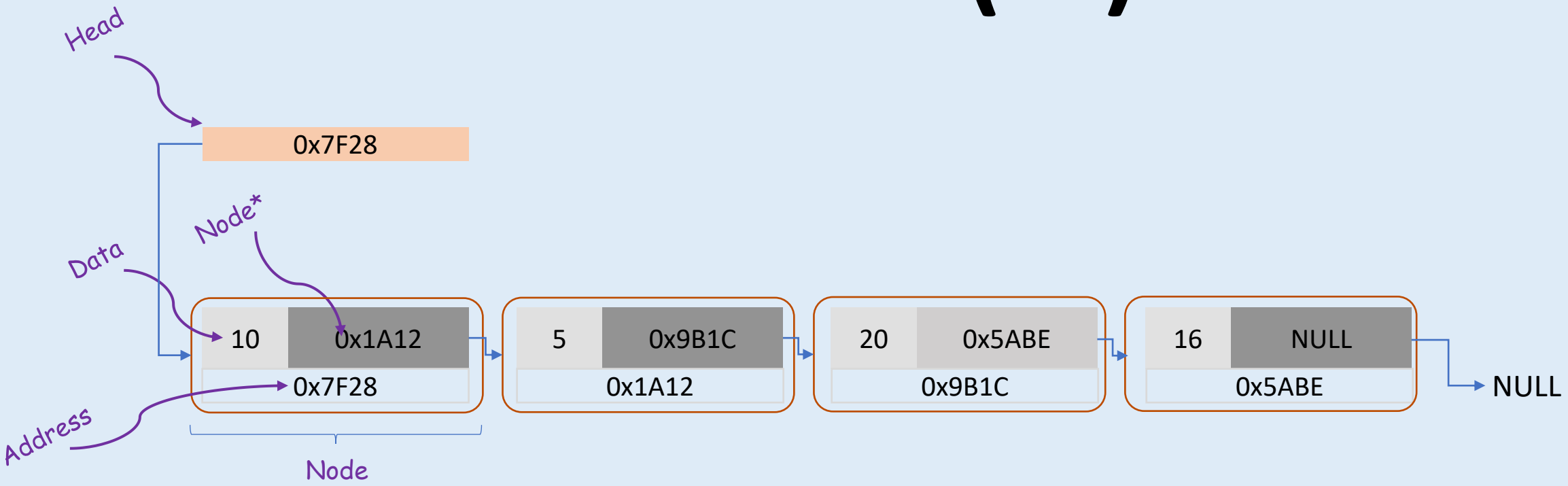
Demo Example

- *Insert: Demonstrate adding a new node, updating pointers.*
- *Search: Show traversing nodes to find a target.*
- *Delete: Find the target, adjust pointers, and remove the node*



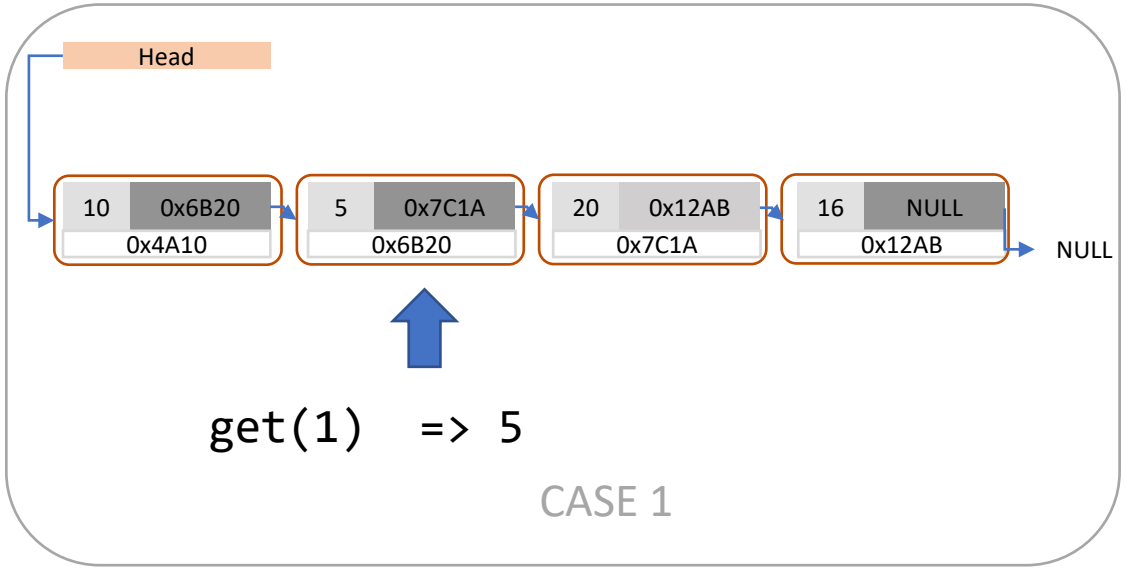
EXPLAIN

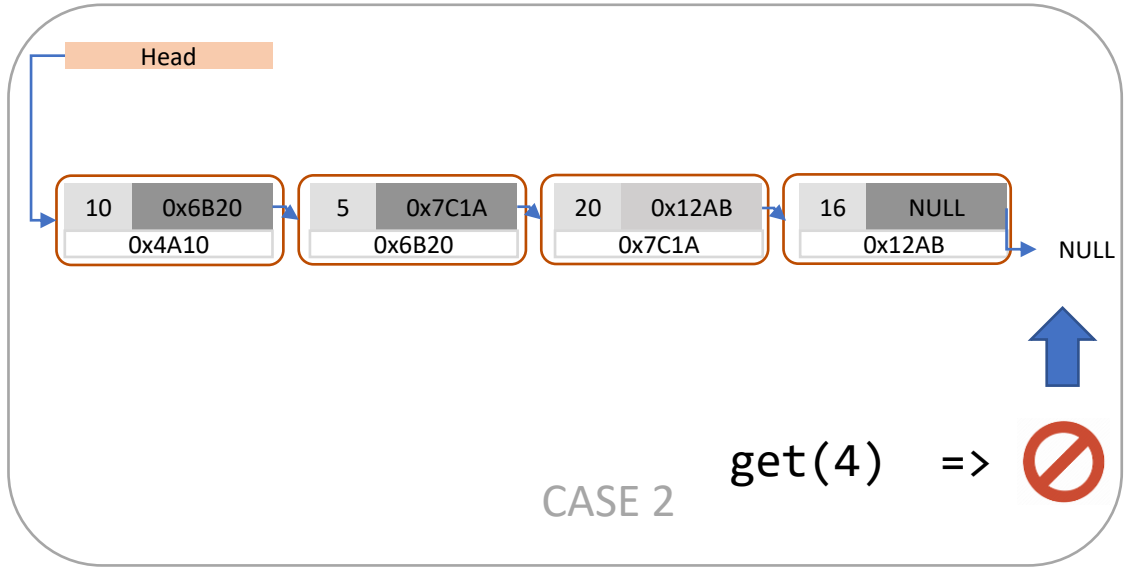
Linked List (LL)



1 int get(index)

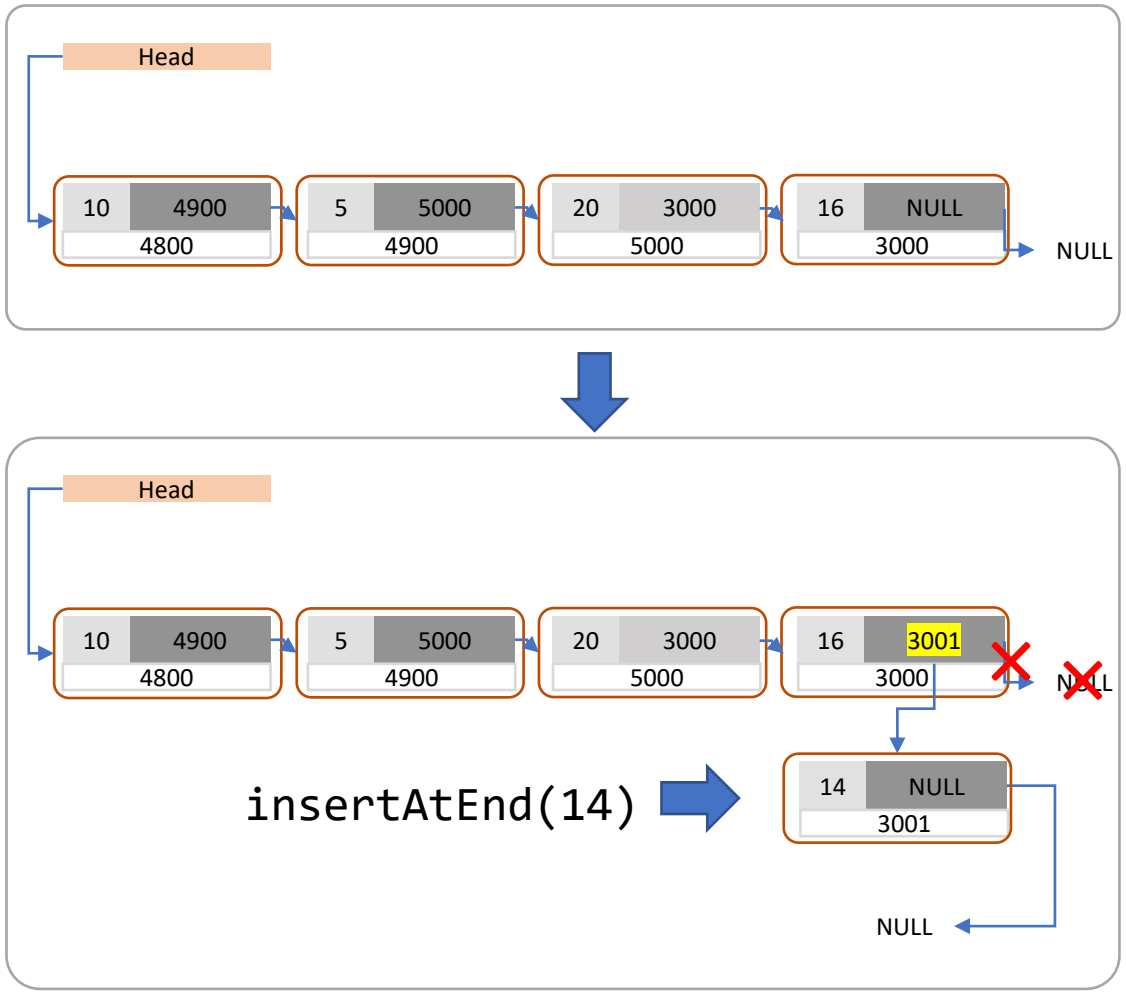
Syntax	int get(int index)
Description	Get the **value of the data** at given index from the list
Precondition	The index must be in the range 0… size-1
Example	`myList = [10, 5, 20, 16]` `int value = myList.get(1)` *-> value should be 5*
Complexity	$O(n)$ The function iterates from the head node to the specified index. In the worst case (when the index is at the end of the list), it traverses $n-1$ nodes, making this step $O(n)$.





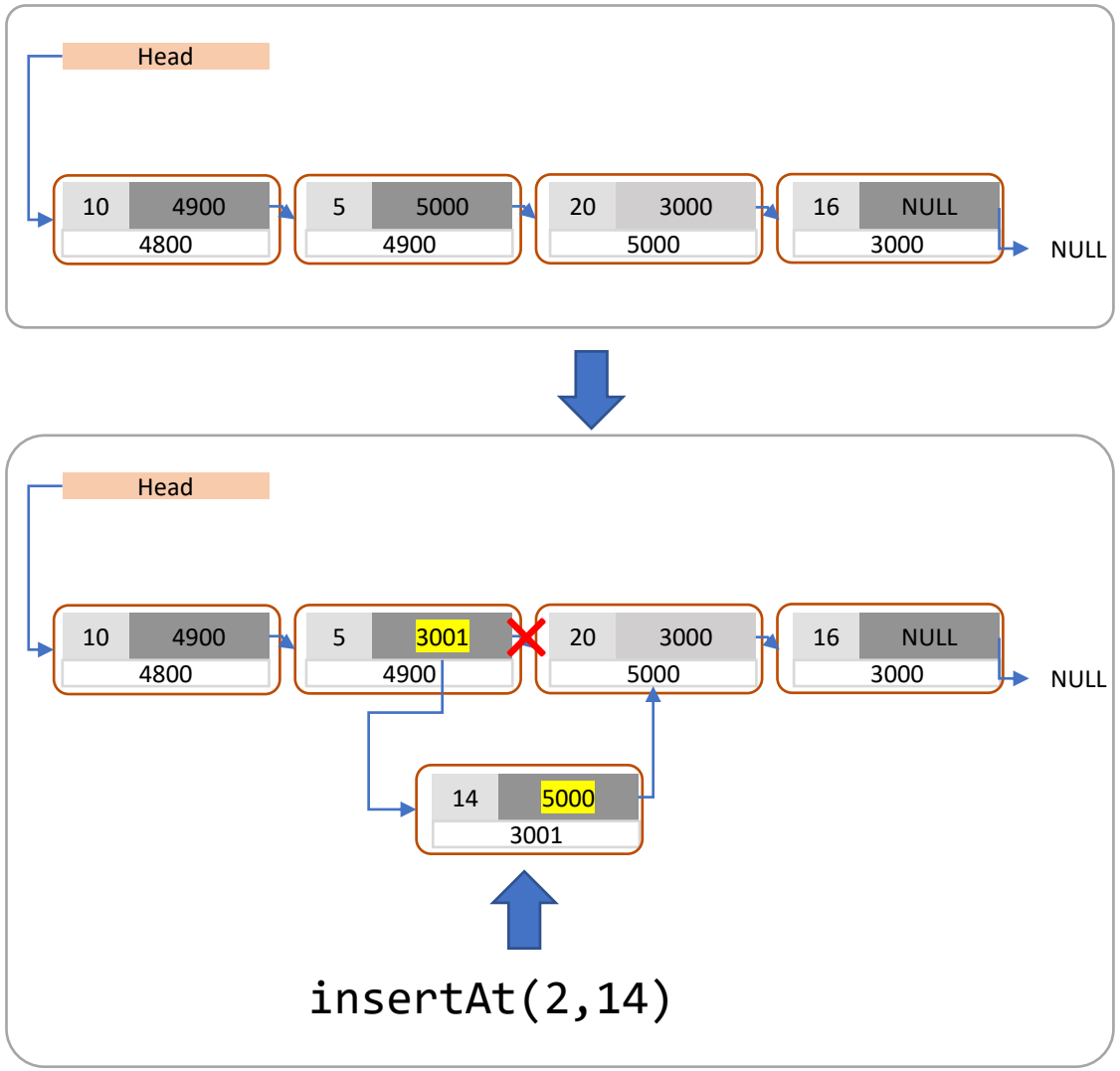
2 void insertAtEnd(value)

Syntax	void insertAtEnd(value) or push(value)
Description	Insert a **value at the end** of the array
Precondition	The index must be in the range 0… size-1
Example	`myList = [10, 5, 20, 16]` `myList.insertAtEnd(14)` *-> myList is now [10, 5, 20, 16, 14]*
Complexity	$O(n)$



3 void insertAt(index,value)

Syntax	int insertAt(int index, int value)
Description	Insert a **value at the given index** of the list
Precondition	index < 0 || index > length
Example	`myList = [10, 5, 20, 16]` `myList.insertAt(2, 14)` *-> myList is now [10, 5, 14, 20, 16]*
Complexity	O(n)



ACTIVITY

10 MIN

Activity !!

The operation **removeAt** *remove a node at a given index in the List*

Q1 - Define the **specifications** of this operation

Syntax	
Description	
Precondition	
Example	
Complexity	

Q2 – Identify different use cases

```
[10,11,--,--]
insertAt(1,12)
[10,12,11,--]
```

```
[10,11,12]
insertAt(1,13)
Array is full
```

CORRECTION

10 MIN

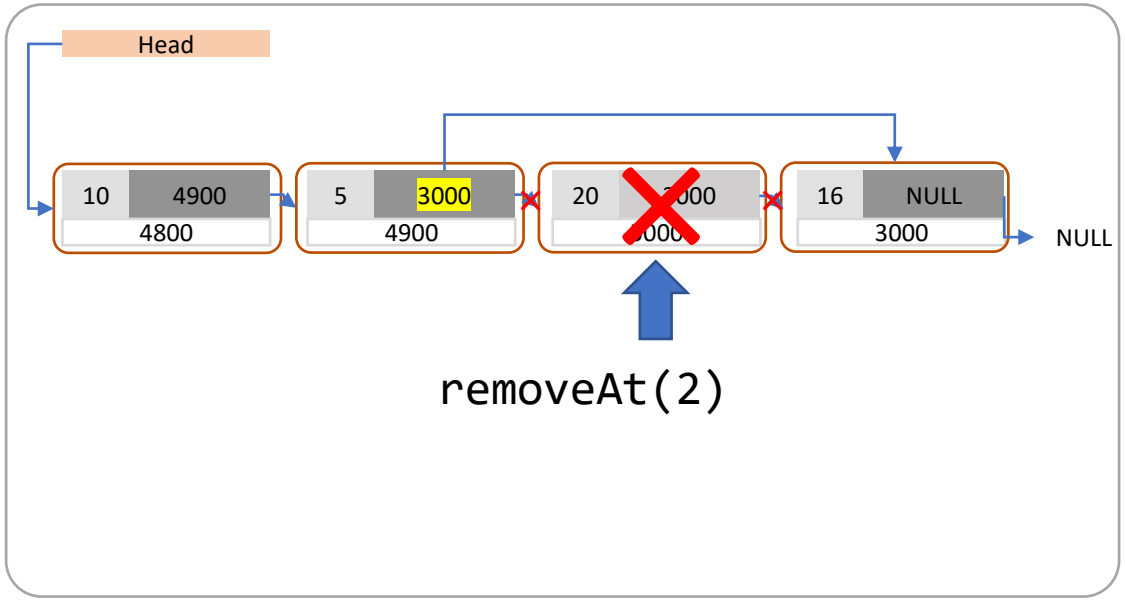
Activity !!

The operation **removeAt** *remove a node at a given index in the List*

Q1 - Define the **specifications** of this operation

Syntax	void removeAt(int index, int value)
Description	remove a **node at given index in** the array
Precondition	index < 0 index >= length
Example	`myList = [10, 5, 20, 16]` `myList.removeAt(2)` *-> myList is now [10, 5, 16]*
Complexity	O(n)

Q2 – Identify different use cases



Double Linked List ?

Resources

To go further…

Follow this playlist to understand the most important data structures

You should know…

- Understand the Concept of **Linked List**
- Differentiate Between **Array** and **Linked List**
- Define the operations of a **Linked List** and their **complexity**

3-2-1 Challenge

- ✓ List three things you **learned** today.
- ✓ List two **questions** you still have.
- ✓ List one aspect of the lesson or topic you **enjoyed**.