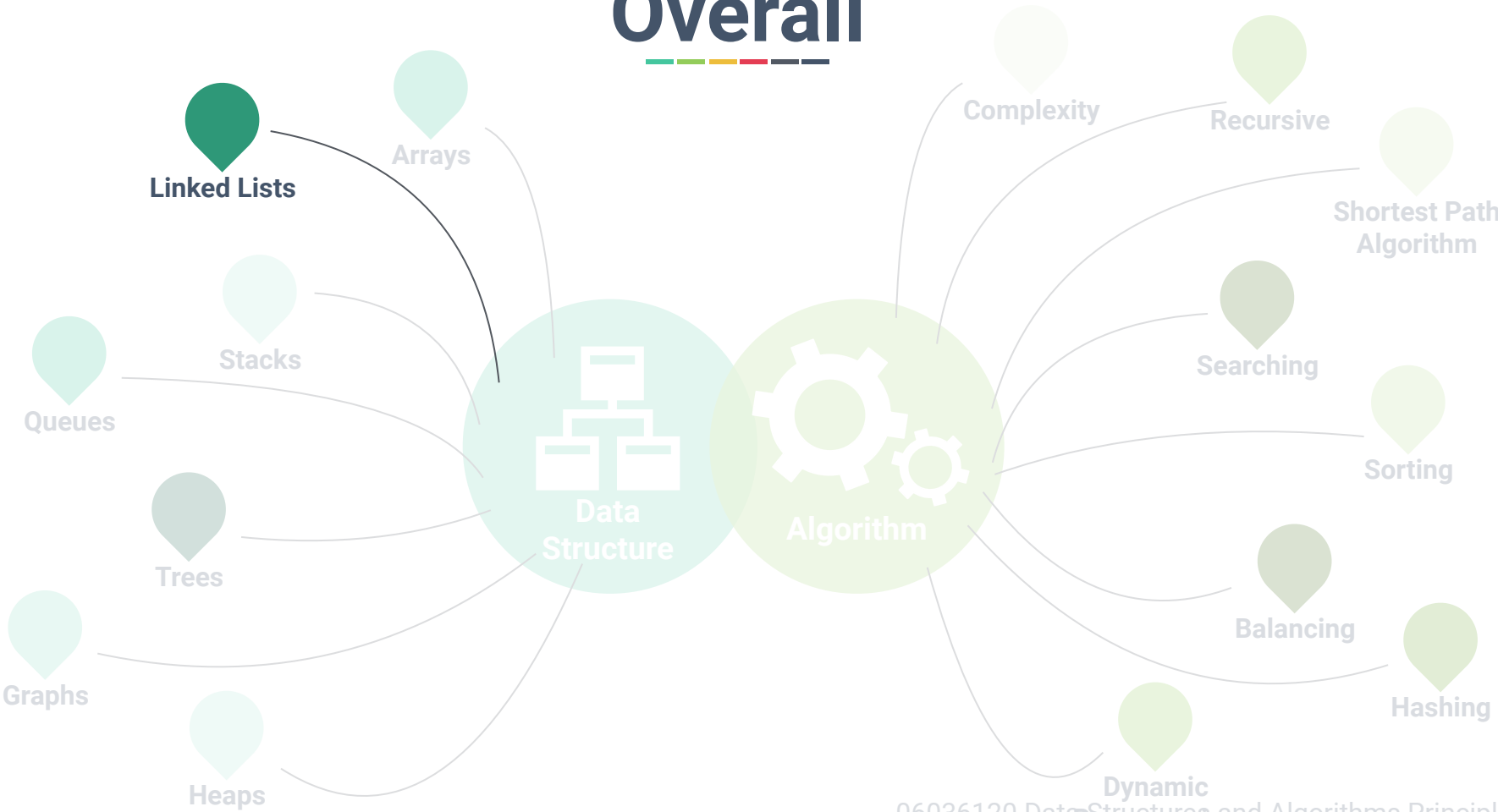
Chapter 5: Linked Lists

Dr. Sirasit Lochanachit

Overall


Linked Lists
Arrays
Stacks
Queues
Trees
Graphs
Heaps
Data Structure
Algorithm
Complexity
Recursive
Shortest Path Algorithm
Searching
Sorting
Balancing
Hashing
Dynamic Programming

Today's Outline

1. What is a Linked List?
2. Singly Linked Lists
    - Traversing
    - Insert a node
    - Delete a node
    - Stack and Queue Implementation

Previously

Python's array-based list

- Stack
- Queue

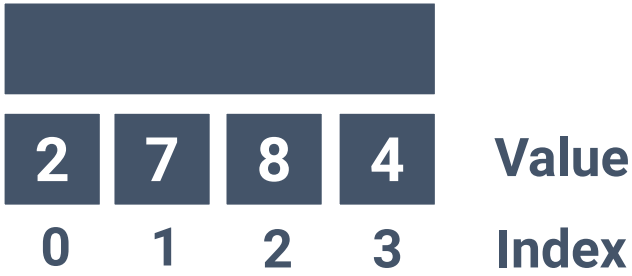
Disadvantages of array:

- Length of array has to be pre-allocated, empty space wasted.
- Adding or removing elements between values in the array is expensive - $O(n)$

Linked Lists

To avoid these limitations, an alternative to array is **linked list**.

Array



Linked lists

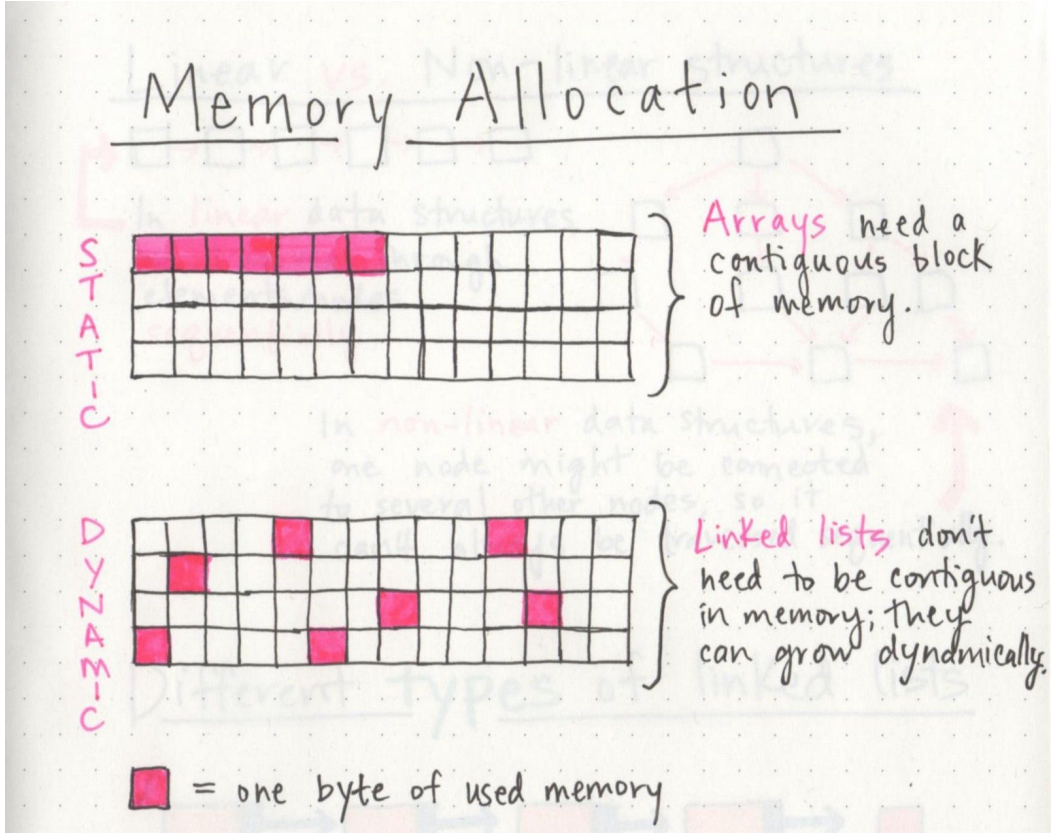
Linked Lists

Static

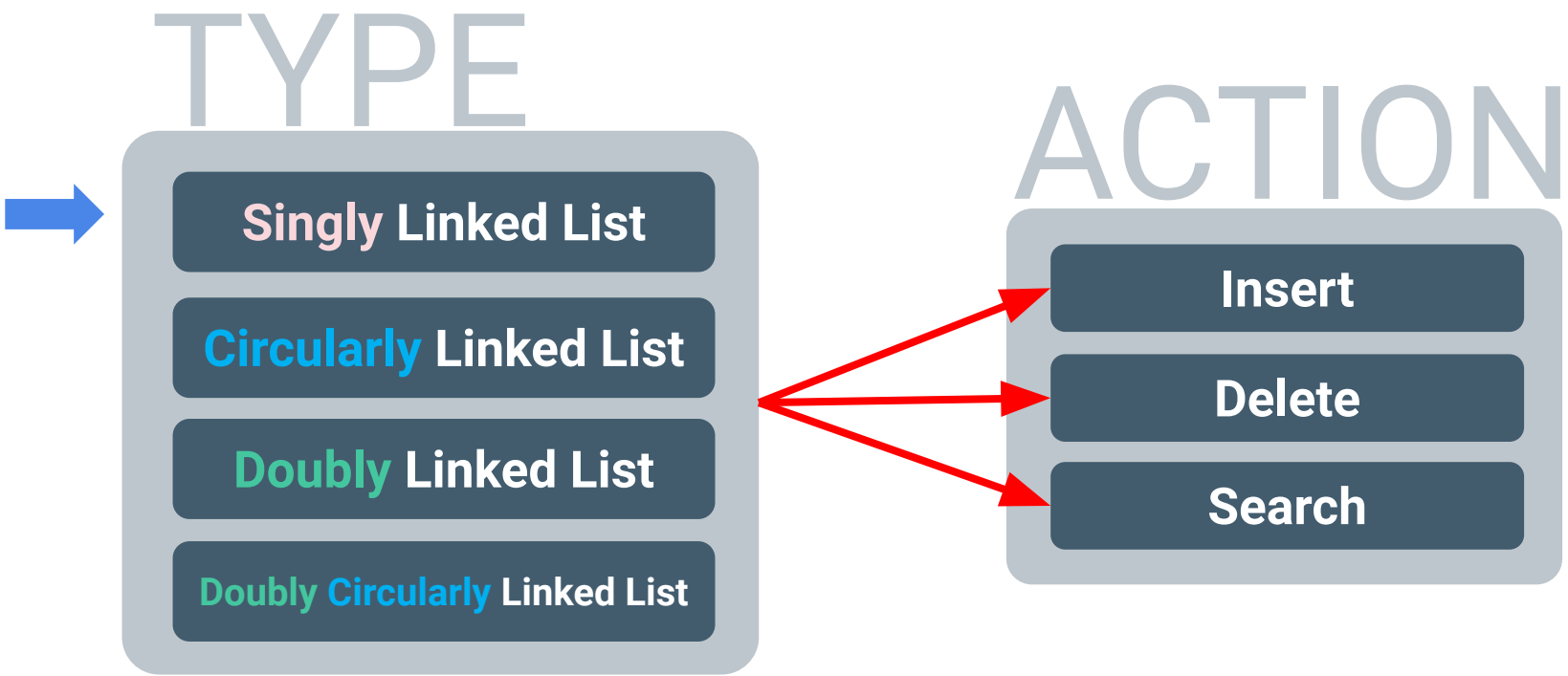
- Pre-allocated
- Fixed size
  - Unable to grow

Dynamic

- Allocated as needed
- Able to grow



Linked Lists



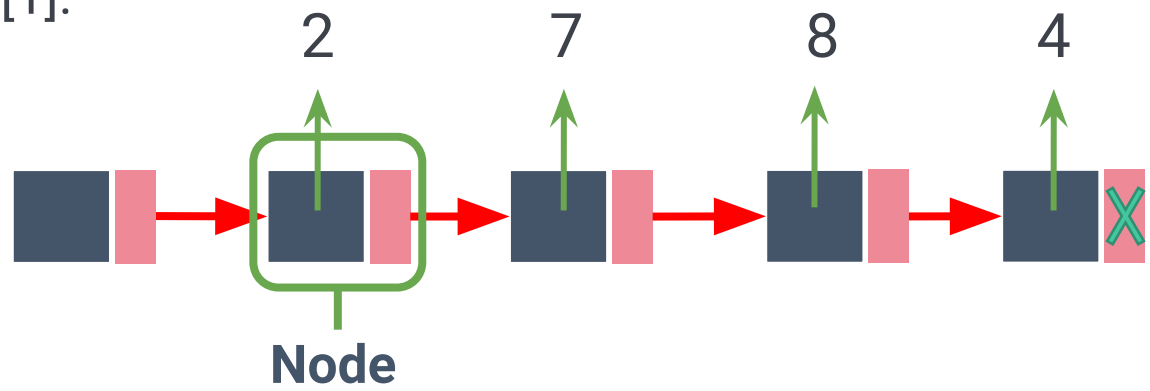
What is a Linked List?

A singly **linked list** is a collection of nodes that form a linear order of a sequence [1].

[1] Michael T. Goodrich et al., Data Structures and Algorithms in Python, 2013

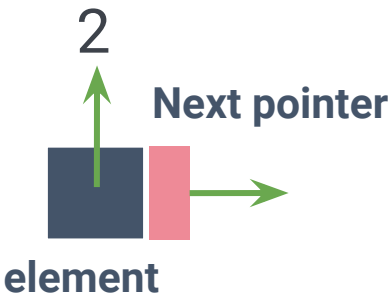
What is a Linked List?

A singly **linked list** is a collection of nodes that form a linear order of a sequence [1].



[1] Michael T. Goodrich et al., Data Structures and Algorithms in Python, 2013

Linked List Node



Linked List Node Structures

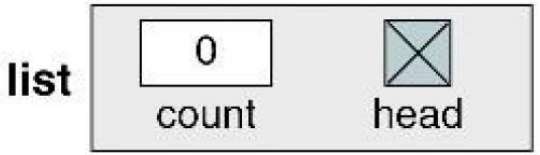
Create a Linked List

1. **Create a header/root node**

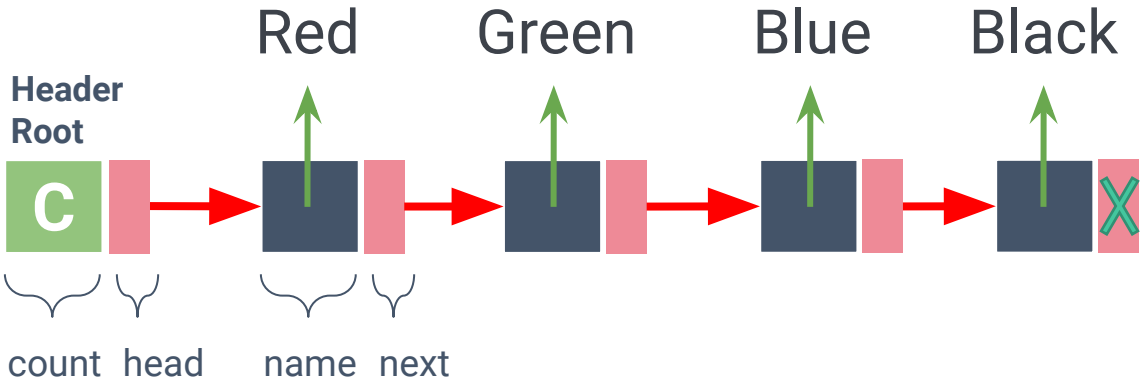
Algorithm createList (list)

1. Allocate a list
2. Set list head to null
3. Set list count to 0

End createList



Singly Linked Lists



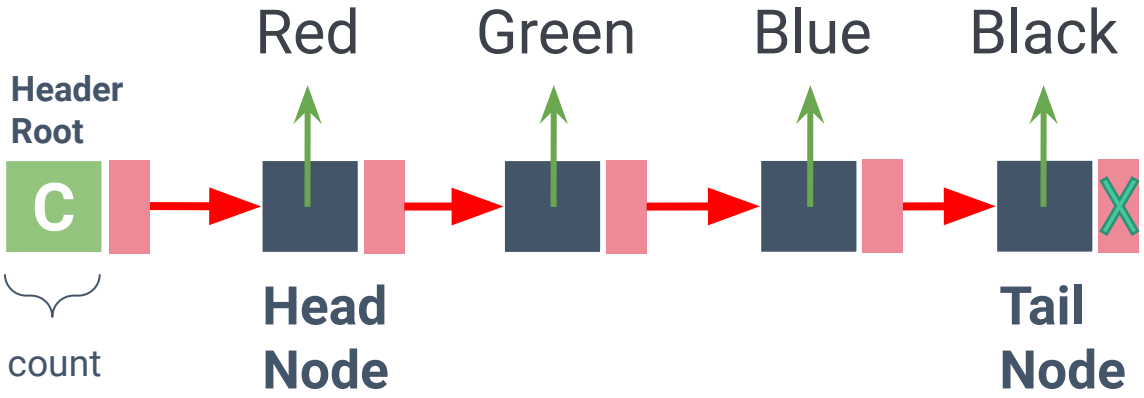
Color

String name

Color next

End Color

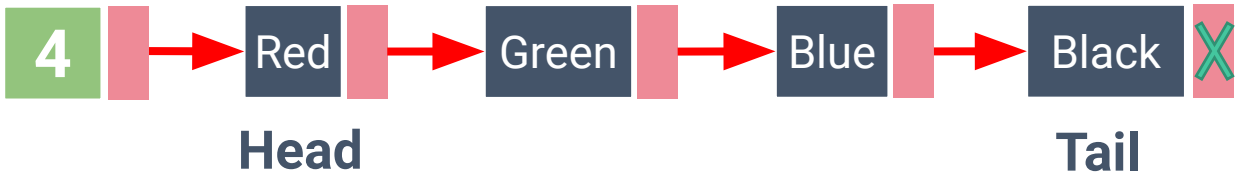
Singly Linked Lists



Linked Lists Examples

Real-life examples of Linked Lists:

Singly Linked Lists



Create a Linked List

2. Create a data/element Node

```
Algorithm createDataNode (d, p)
    colorNew = allocate(Color)
    name = d
    next = p
    return colorNew
End createDataNode
```

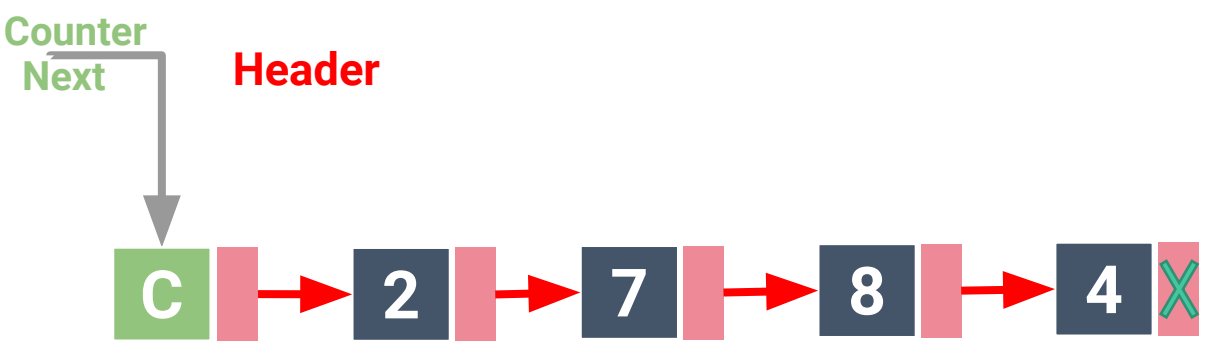
Traversing Singly Linked Lists

Address/ Byte#	Value
6000	4
6001	6002
6002	2
6003	6008
6004	8
6005	6012
6006	
6007	
6008	7
6009	6004
6010	
6011	
6012	4
6013	None

Suppose that it takes 1 byte to store an integer.

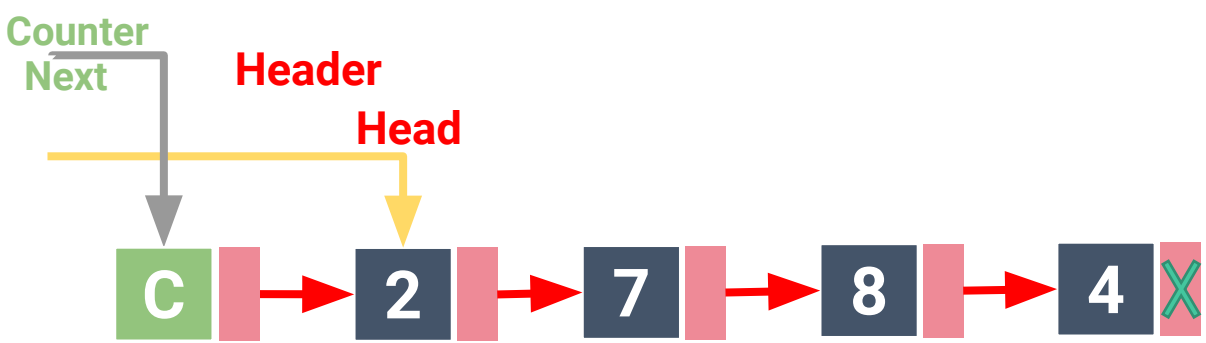
Traversing Singly Linked Lists

Address/ Byte#	Value
6000	4
6001	6002
6002	2
6003	6008
6004	8
6005	6012
6006	
6007	
6008	7
6009	6004
6010	
6011	
6012	4
6013	None



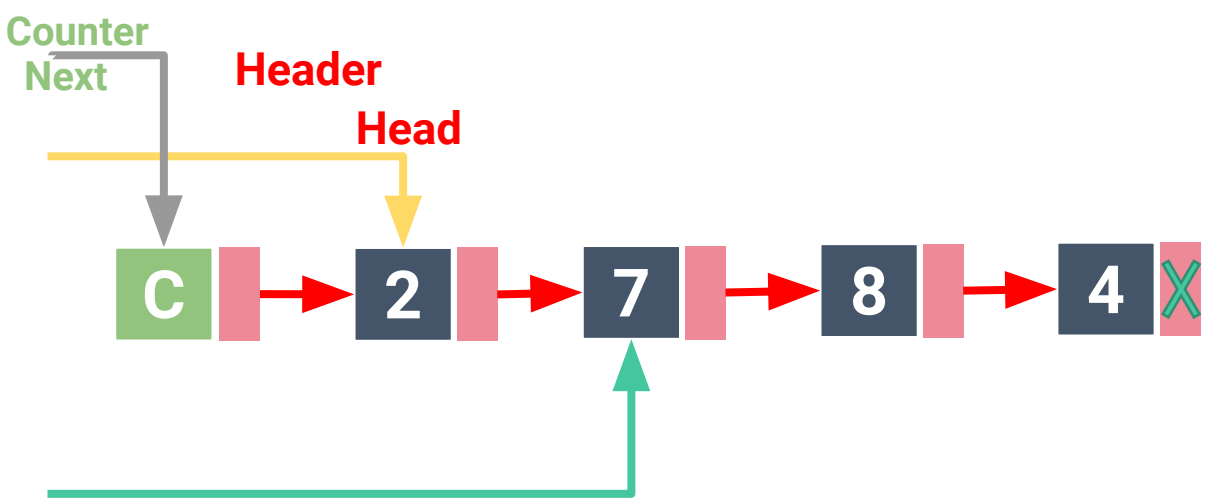
Traversing Singly Linked Lists

Address/ Byte#	Value
6000	4
6001	6002
6002	2
6003	6008
6004	8
6005	6012
6006	
6007	
6008	7
6009	6004
6010	
6011	
6012	4
6013	None



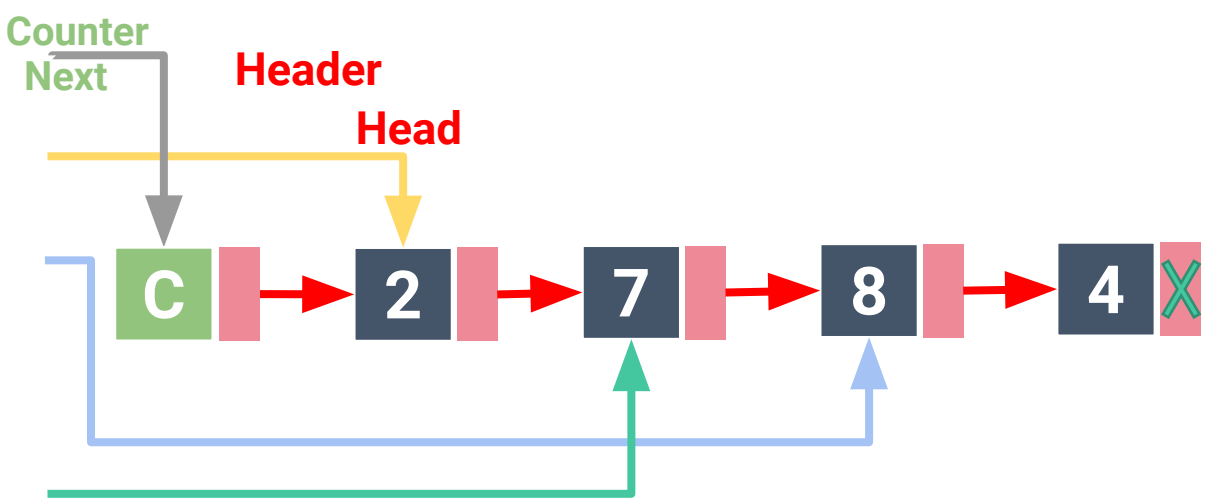
Traversing Singly Linked Lists

Address/ Byte#	Value
6000	4
6001	6002
6002	2
6003	6008
6004	8
6005	6012
6006	
6007	
6008	7
6009	6004
6010	
6011	
6012	4
6013	None

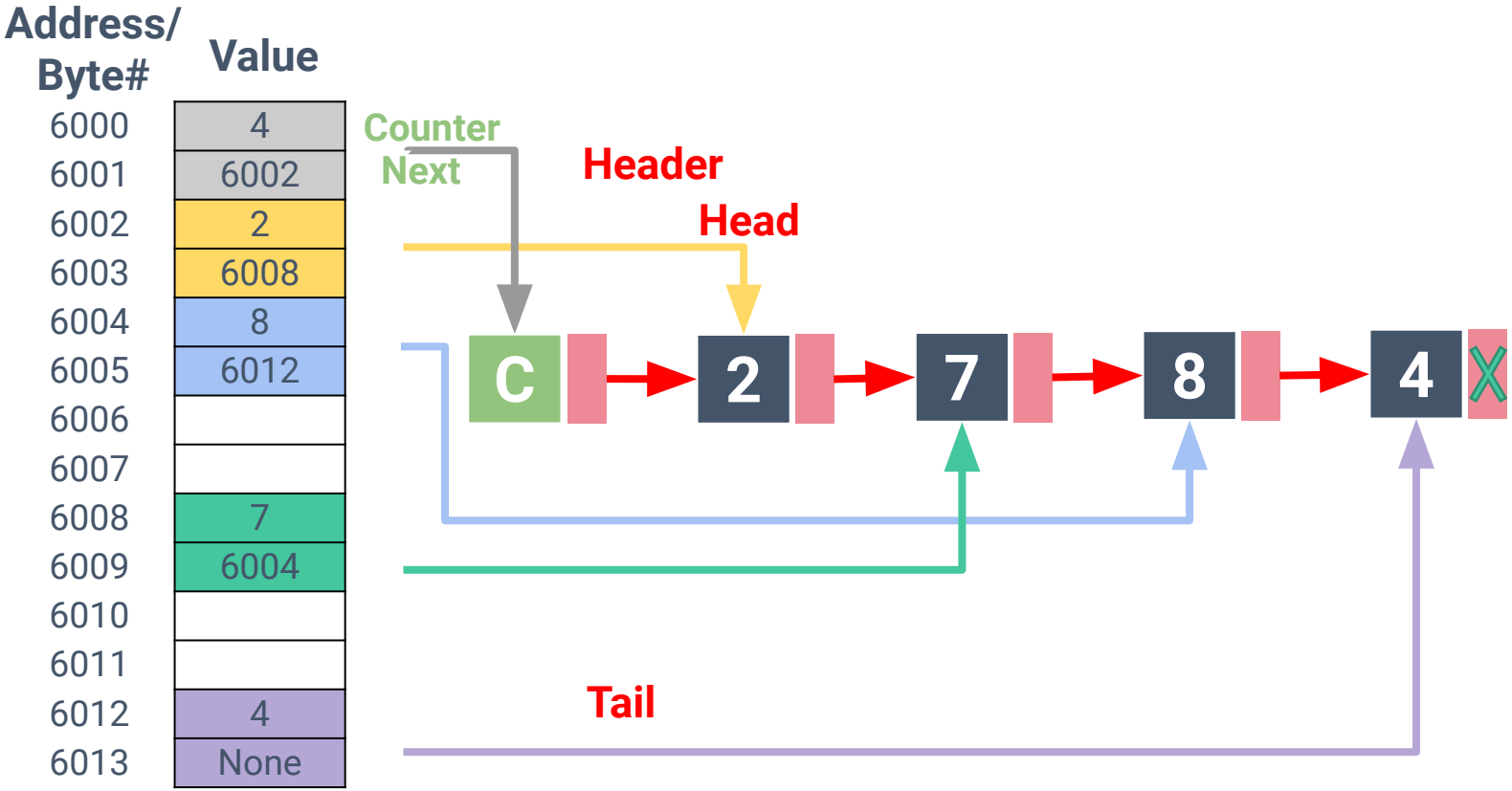


Traversing Singly Linked Lists

Address/ Byte#	Value
6000	4
6001	6002
6002	2
6003	6008
6004	8
6005	6012
6006	
6007	
6008	7
6009	6004
6010	
6011	
6012	4
6013	None

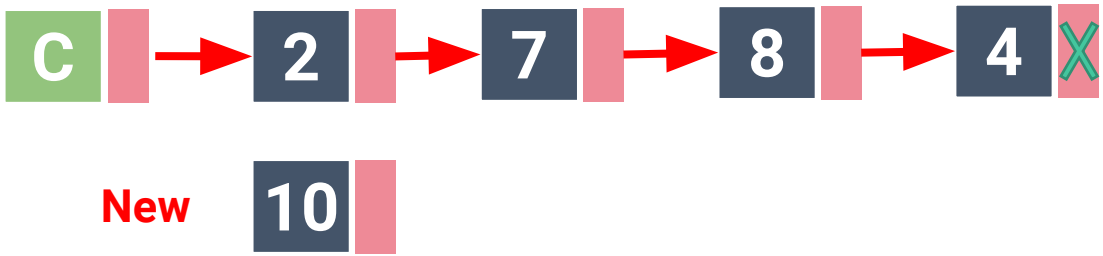


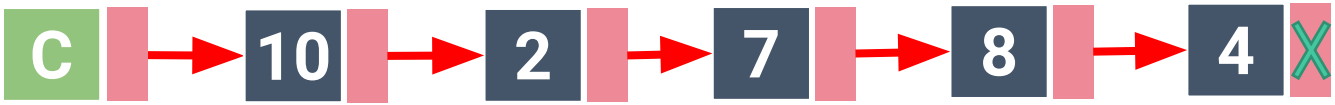
Traversing Singly Linked Lists



Singly Linked Lists

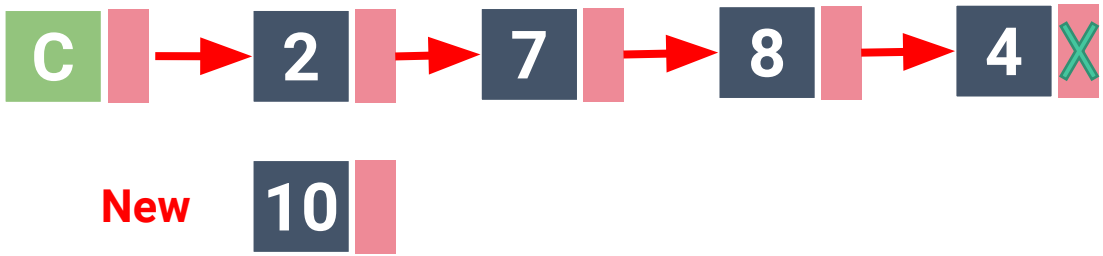
Insert at the head of the list





Singly Linked Lists

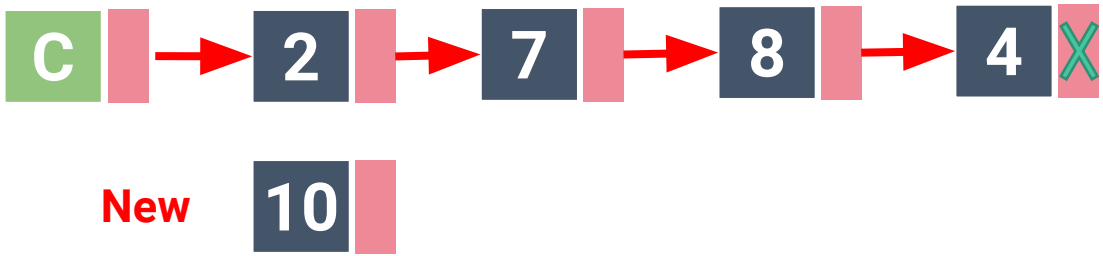
Insert at the head of the list



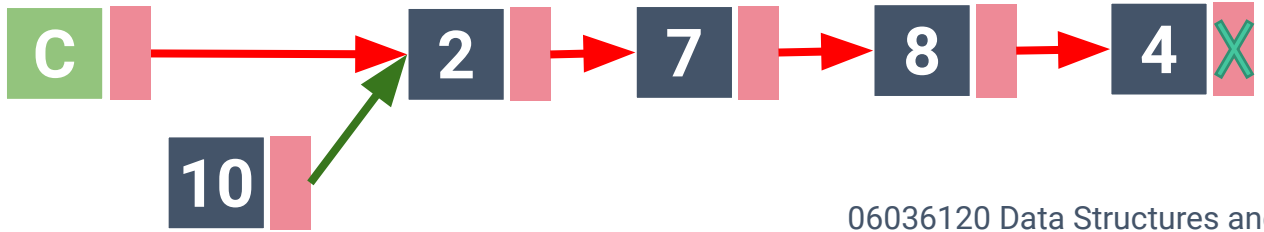
Singly Linked Lists

Insert at the head of the list

Step 1



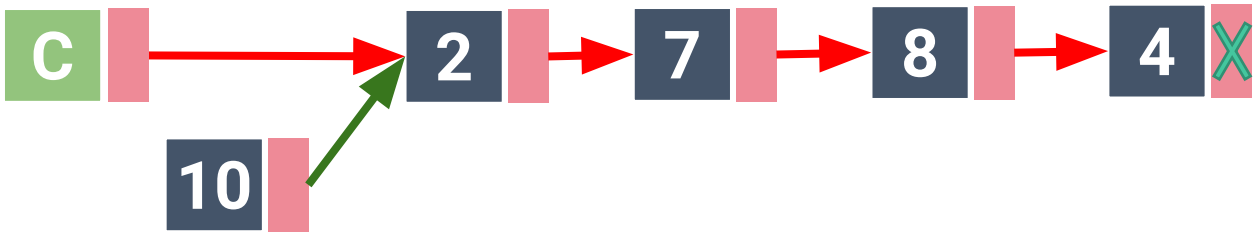
Step 2



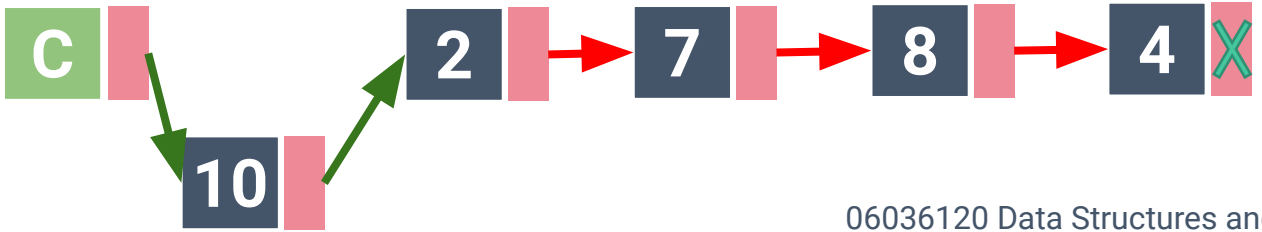
Singly Linked Lists

Insert at the head of the list

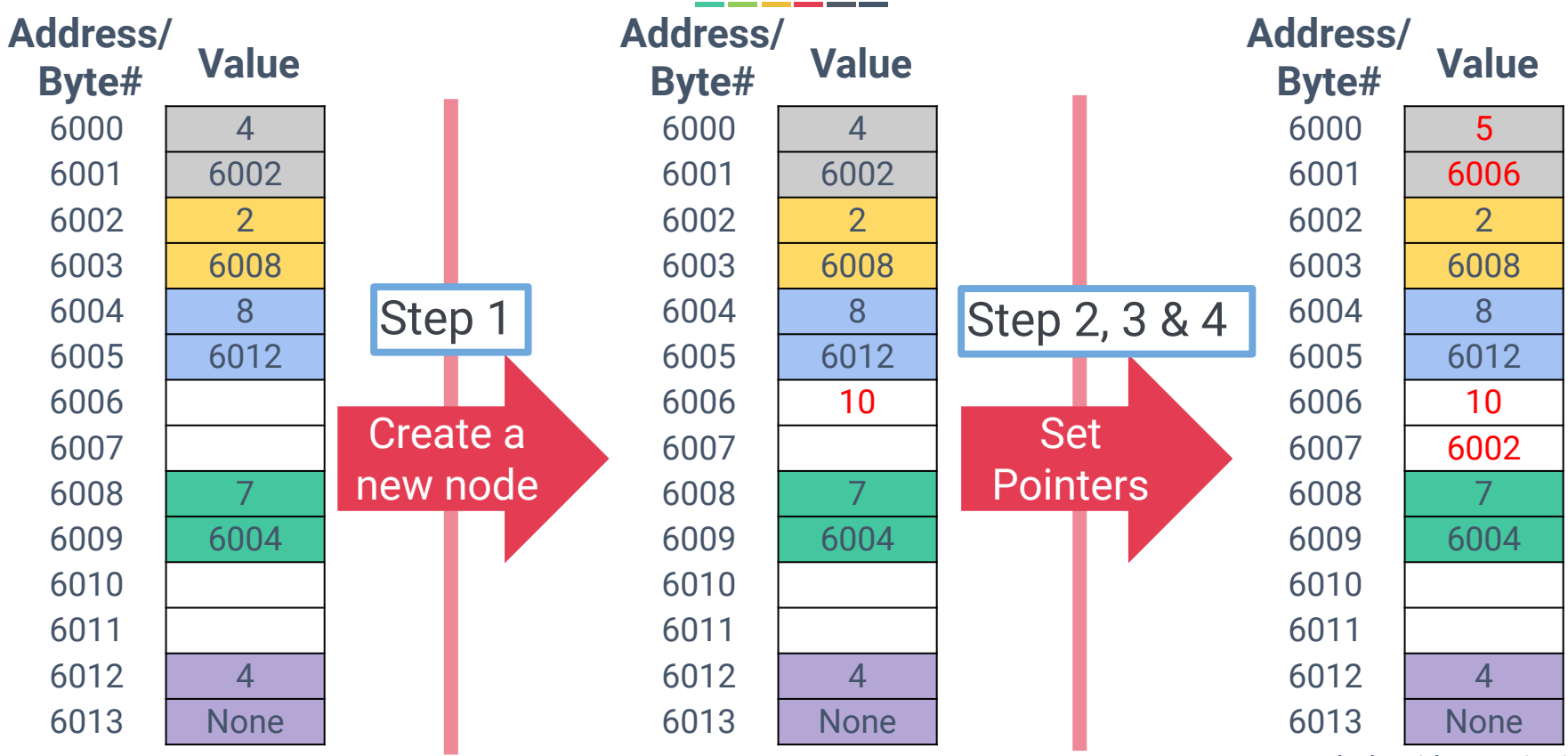
Step 2



Step 3 & 4

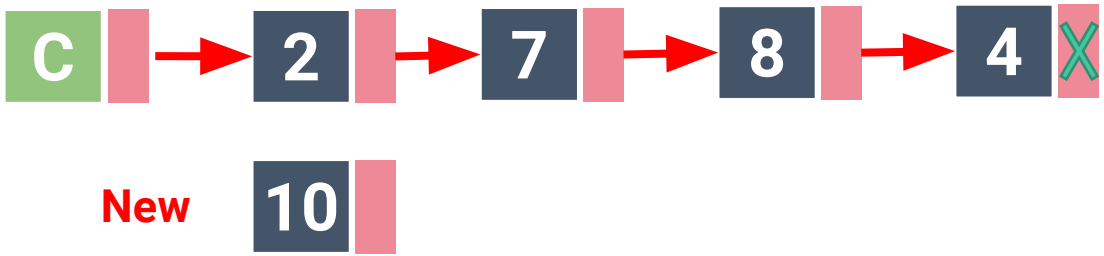


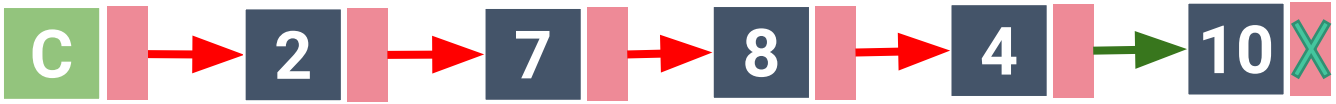
Singly Linked Lists



Singly Linked Lists

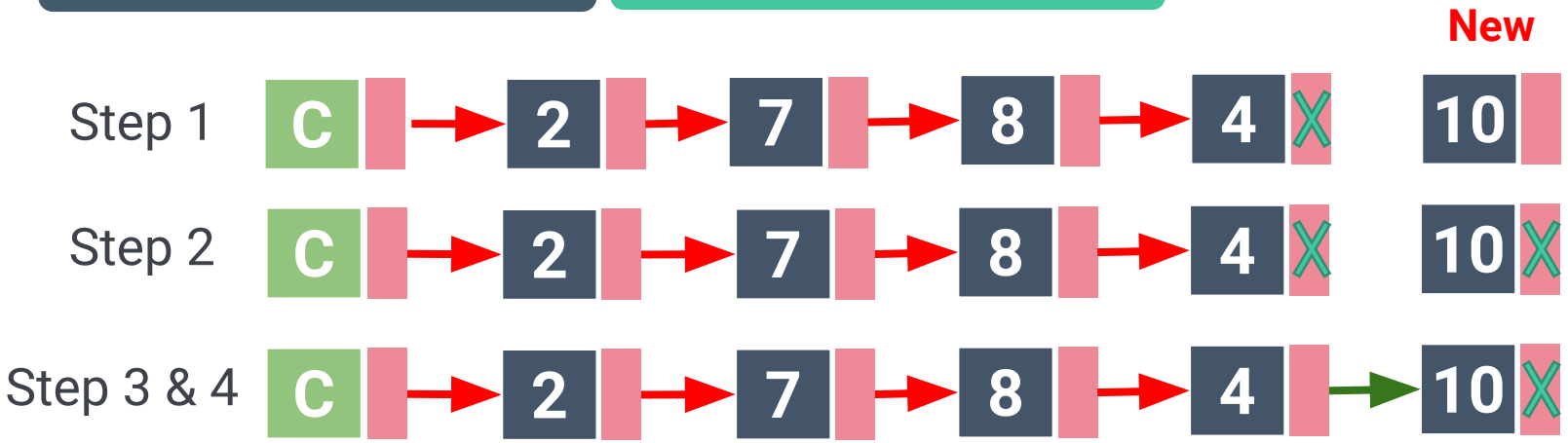
Insert at the tail of the list



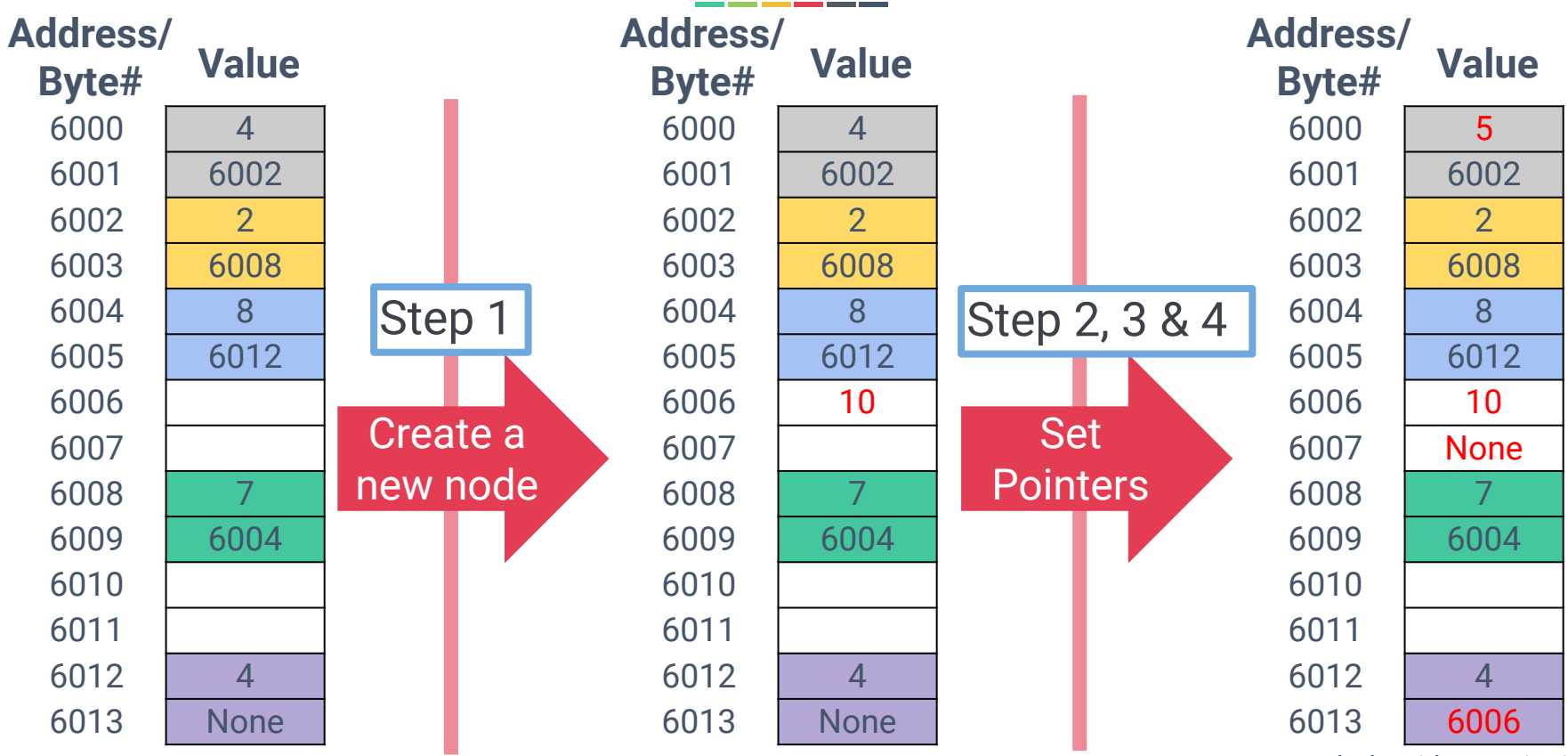


Singly Linked Lists

Insert at the tail of the list

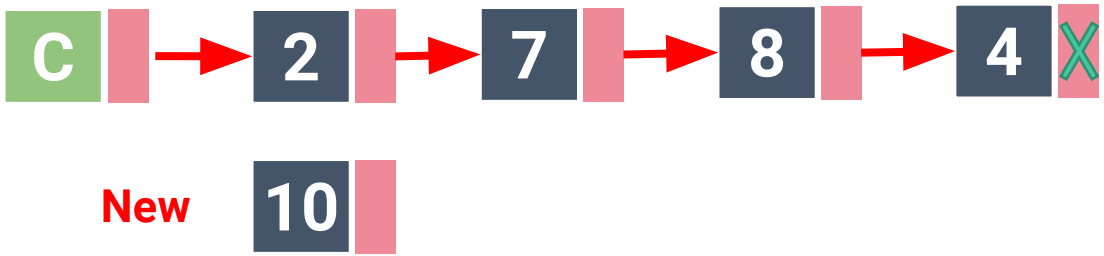


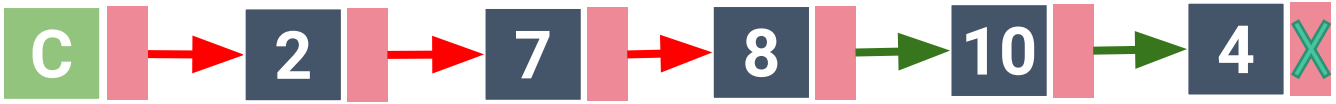
Singly Linked Lists



Singly Linked Lists

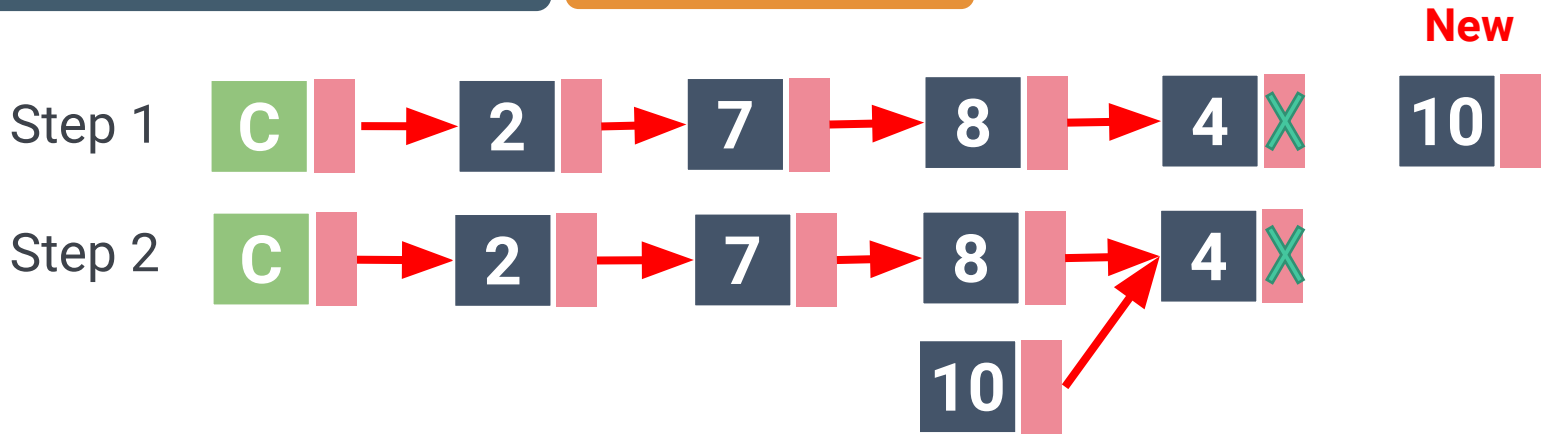
Insert between nodes





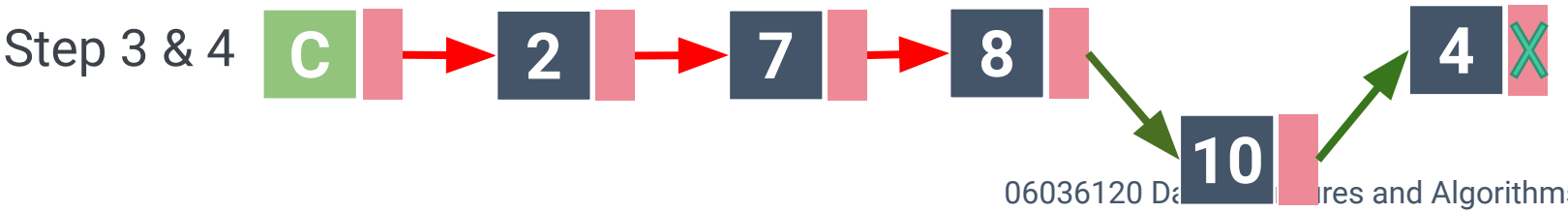
Singly Linked Lists

Insert between nodes



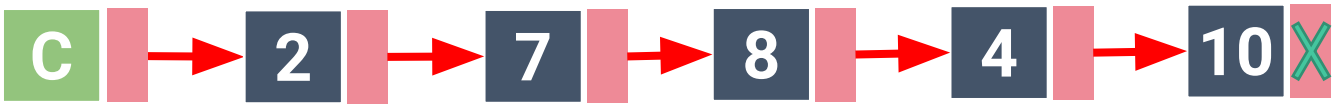
Singly Linked Lists

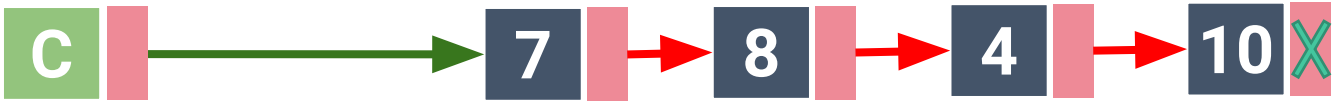
Insert between nodes



Singly Linked Lists

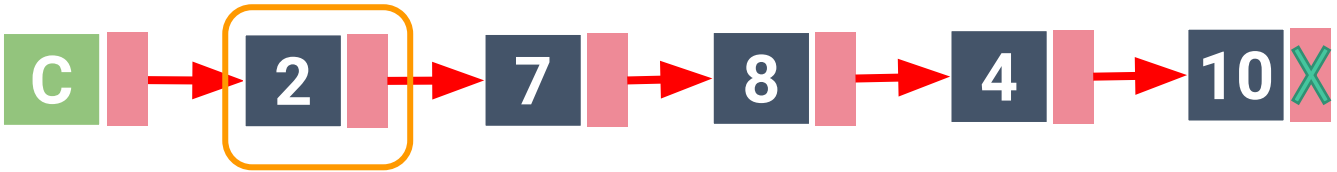
Delete the head node



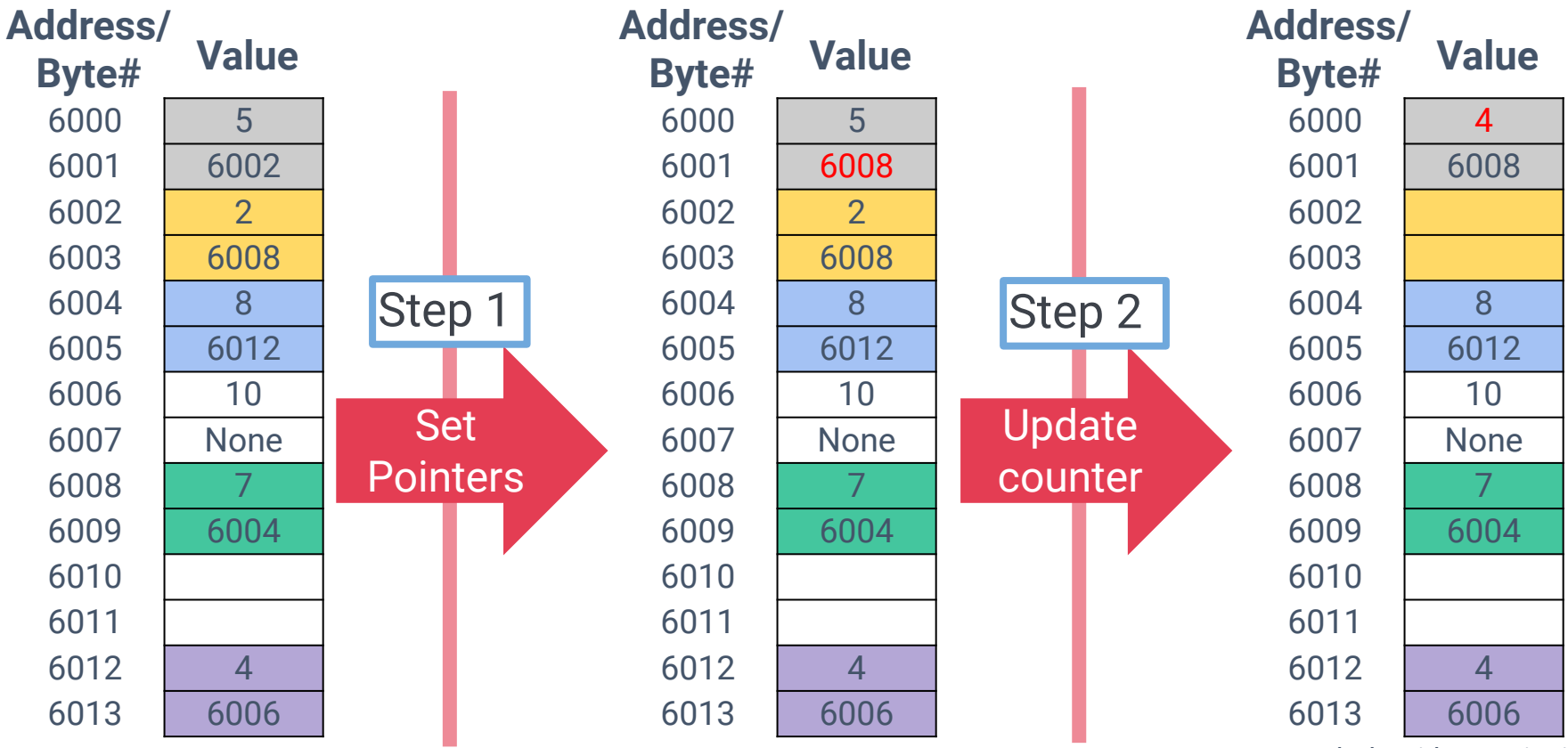


Singly Linked Lists

Delete the head node

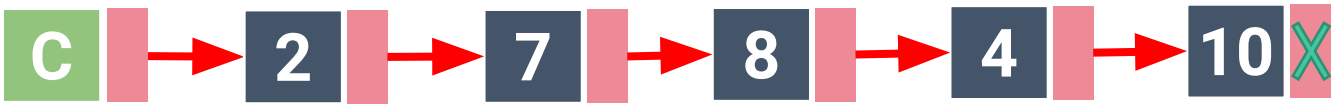


Singly Linked Lists



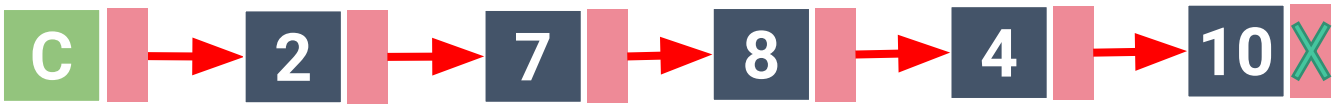
Singly Linked Lists

Delete the tail node

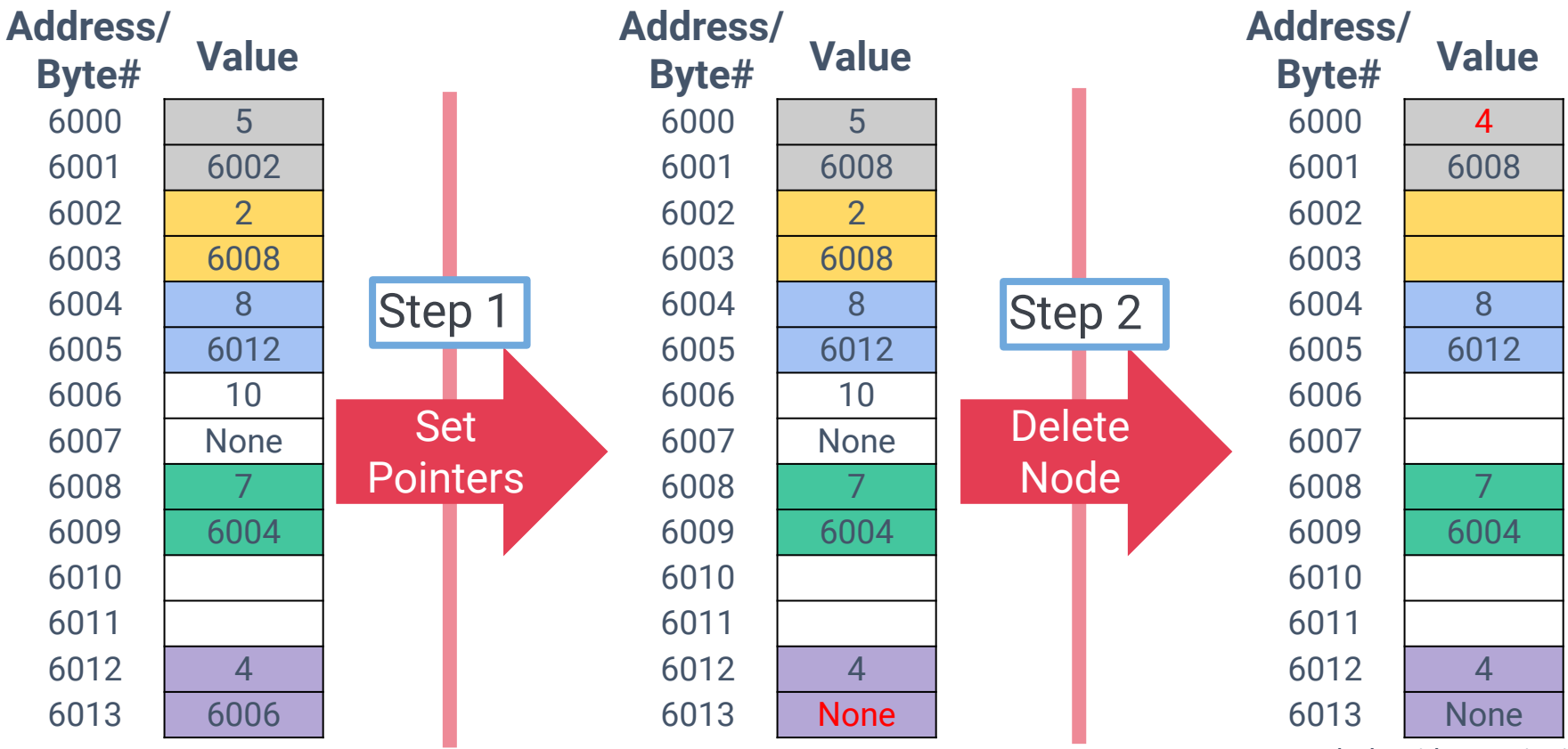


Singly Linked Lists

Delete the tail node



Singly Linked Lists



Singly Linked Lists

Delete between nodes

Step 1

Step 2

Step 3

Singly Linked Lists: Stacks

How to Implement a Stack?

Array!!

and

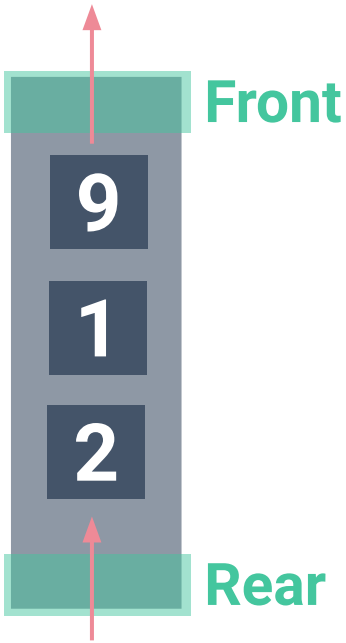
Linked Lists!!

- **Singly Linked Lists**

Asymptotic Performance

Operation	Running Time - Array	Running Time - Singly Linked List
S.push(element)	$O(1)$	
S.pop()	$O(1)$	
S.top()	$O(1)$	
S.is_empty()	$O(1)$	
len(S)	$O(1)$	

Singly Linked Lists: Queues



How to Implement a Queue?

Array!!

and

Linked Lists!!

- **Singly Linked Lists**

Asymptotic Performance

Operation	Running Time - Array	Running Time - Singly Linked List
Q.enqueue(e)	$O(1)$ or $O(n)$	
Q.dequeue()	$O(1)$ or $O(n)$	
Q.first()	$O(1)$	
Q.is_empty()	$O(1)$	
len(Q)	$O(1)$	