Singly Linked Lists

Lesson 4.1

Learning Objectives

LO 4.1.1   **Create** and **initialize** singly linked lists
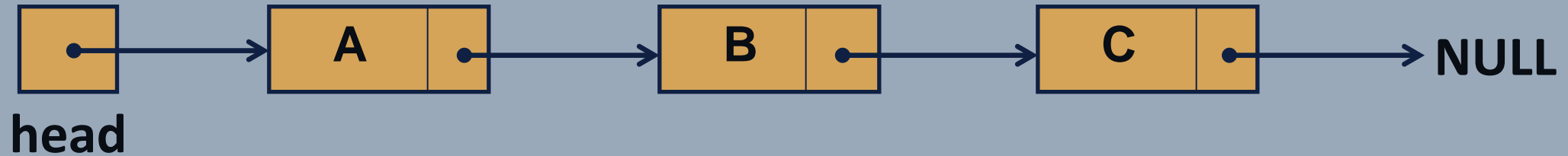
LO 4.1.2   **Perform** insertion and deletion of elements in a singly linked list

Linked List

- A **linked list** is a linear data structure which can change the size of its contents during execution

- It acts like an array as a storage, which are arranged in index locations, yet, you cannot do random access
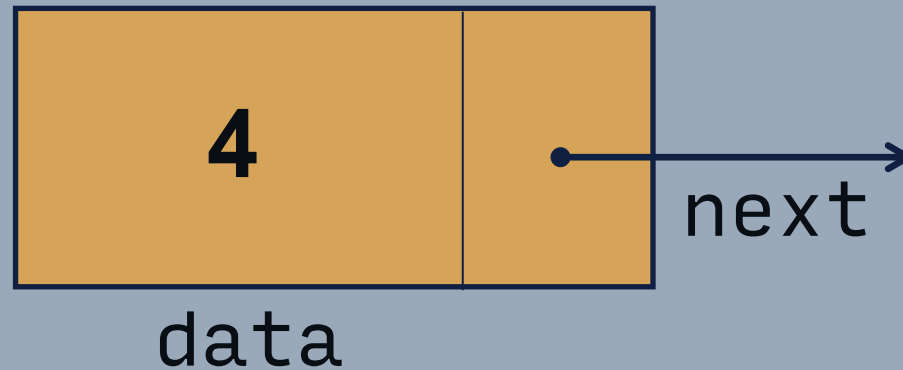
Singly Linked List

- A **singly linked list** is a linked list that only contains 1 pointer in its basic unit and is pointing only to the next node in the sequence of nodes

Basic Units of a Singly Linked List

- A *singly linked list node* contains only 2 parts:

  —The **data** which stores the value of that node; and

  —The **next** pointer that points to the address of the next node in the list

Basic Units of a Singly Linked List

- To make a list of singly linked list nodes, the nodes must be interconnected by linking the next pointers sequentially to the nodes

- The first node must be pointed by a node pointer named head to facilitate the reference of the list when it will be utilized
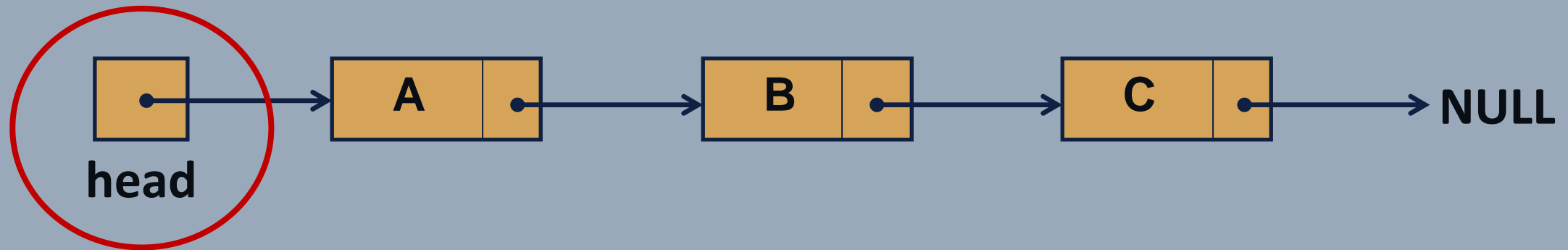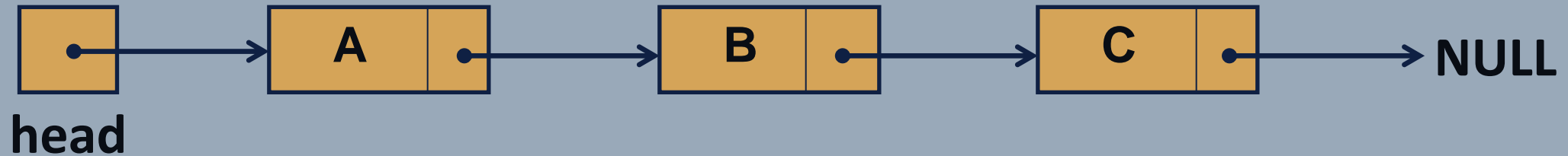
Linked List

- A **linked list** is a data structure which can change the size of its contents during execution

  —Successive elements, known as *nodes*, are connected by pointers

  —The last node points to **NULL**

  —It can grow/shrink in size during execution of a program

  —Addresses of nodes does not require contiguous memory arrangement

# Strengthening the Learning Objectives

LO 4.1.1   Create and initialize singly linked lists

Create a singly linked list with the same configuration below using C structures then display the contents of the linked list by traversing the list using only 1 pointer variable.

LO 4.1.2 Perform insertion and deletion of elements in a singly linked list

On the singly linked list you have just created, insert the data X between B and C, then delete A.

before

after
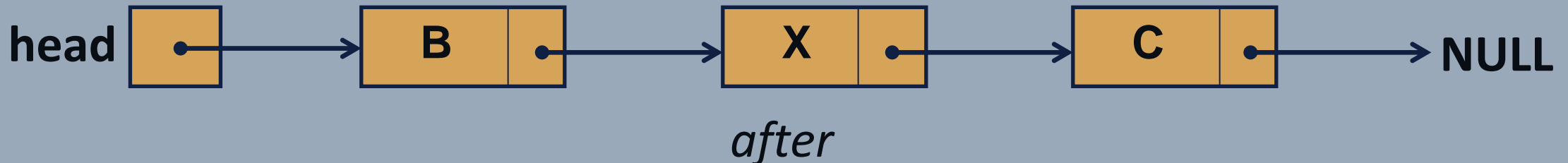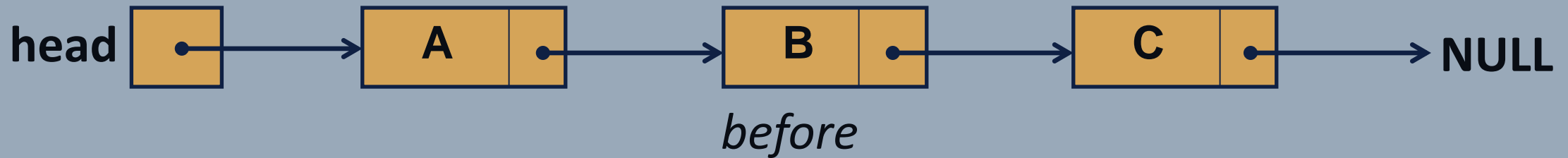
LO 4.1.1   Create and initialize singly linked lists

Create a function

`SINGLYLINKEDLIST createList()`

that will return a pointer pointing to a singly linked list structure with no singly linked list node.

LO 4.1.2 Perform insertion and deletion of elements in a singly linked list

Create a function

```
void addFirst(SINGLYLINKEDLIST list, int data)
```

where **list** is a pointer to an existing singly linked list structure of integer data and **data** is to be inserted at the first/front part of the list.

LO 4.1.2 Perform insertion and deletion of elements in a singly linked list

Create a function

```
void addLast(SINGLYLINKEDLIST list, int data)
```

where **list** is a pointer to an existing singly linked list structure of integer data and **data** is to be inserted at the last/rear part of the list.

LO 4.1.2 Perform insertion and deletion of elements in a singly linked list

Create a function

`void deleteData(SINGLYLINKEDLIST list, int data)`

where **`list`** is a pointer to an existing singly linked list structure of integer data and **`data`** is the element to be deleted in the list.