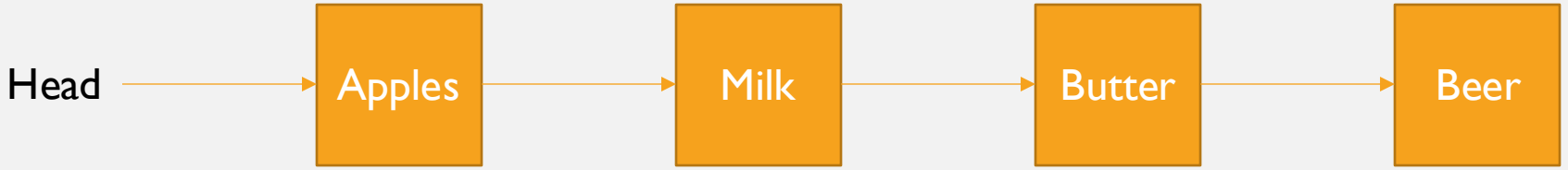
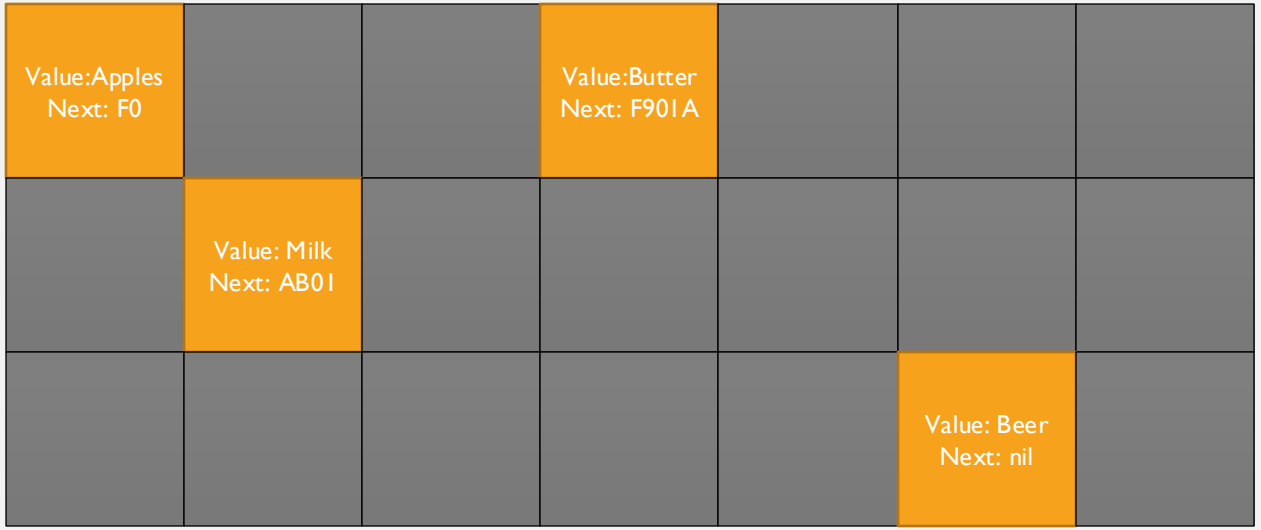
# LINKED LISTS

# LINKED LISTS

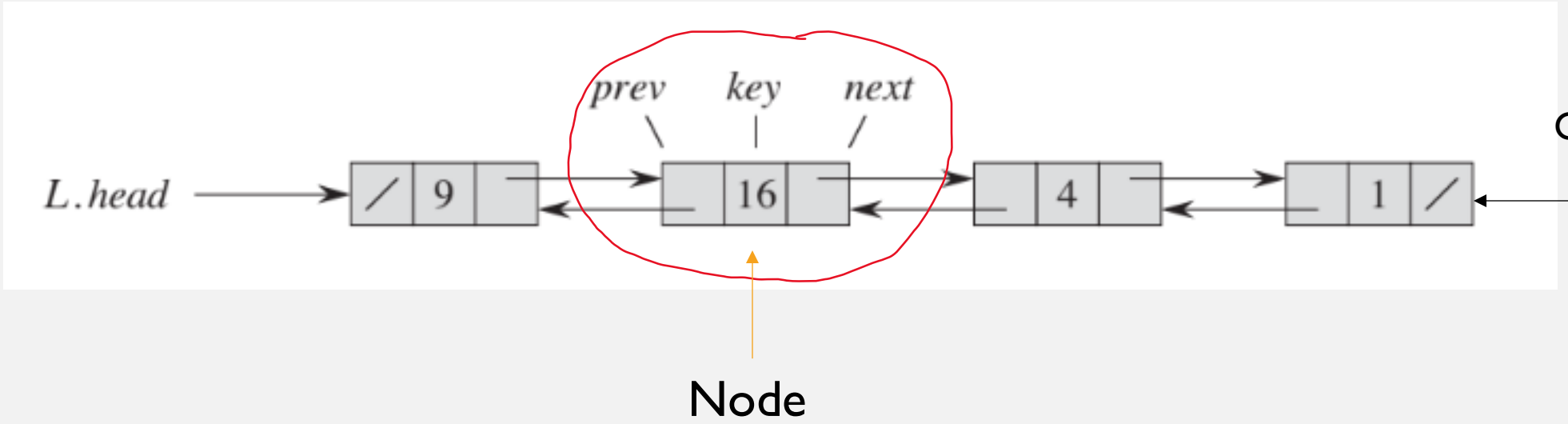
- Objects are arranged in linear order, determined by a pointer (also known as reference) in each of them.



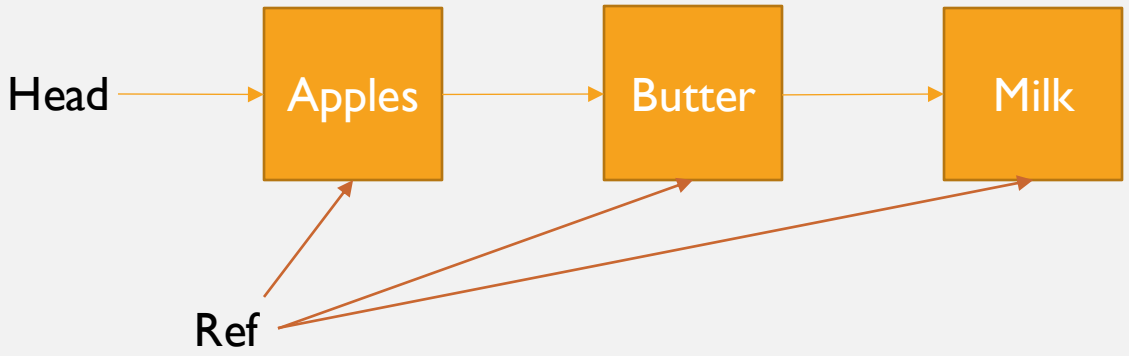



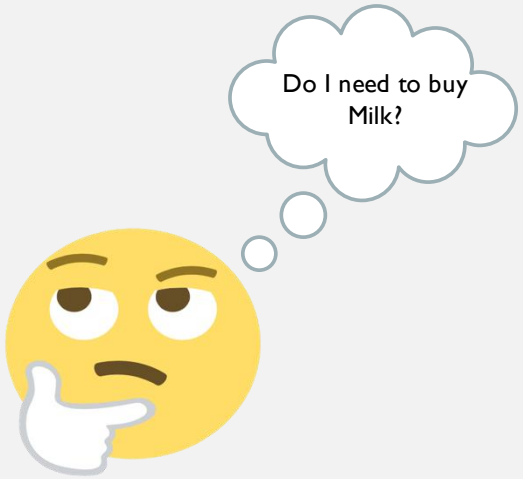
# DOUBLY LINKED LISTS

- Each object bears two pointers: one pointing to the next element, and another to the previous.



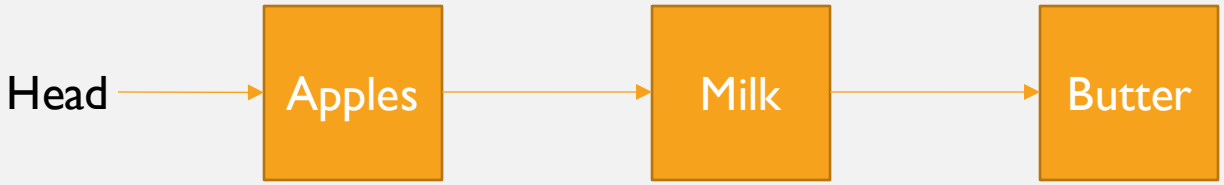
# SEARCH



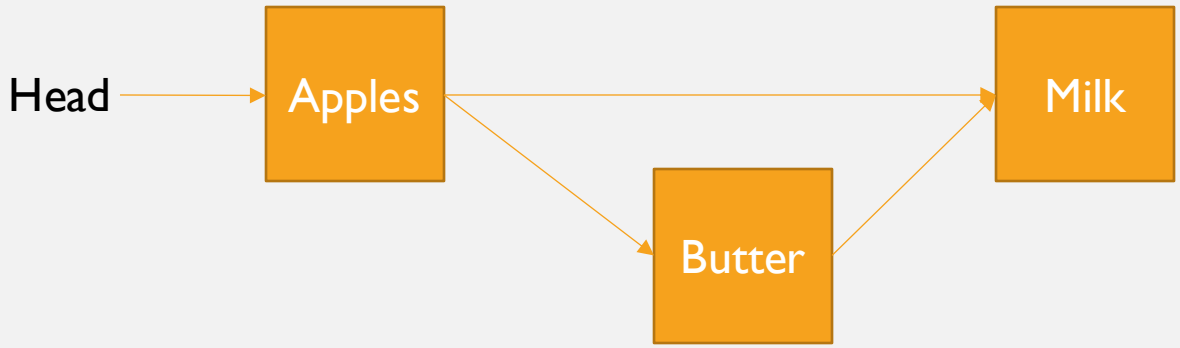

Exercise: Code a function to search for an element in a linked list. It should return the containing node if it's found, or None otherwise.

# APPEND



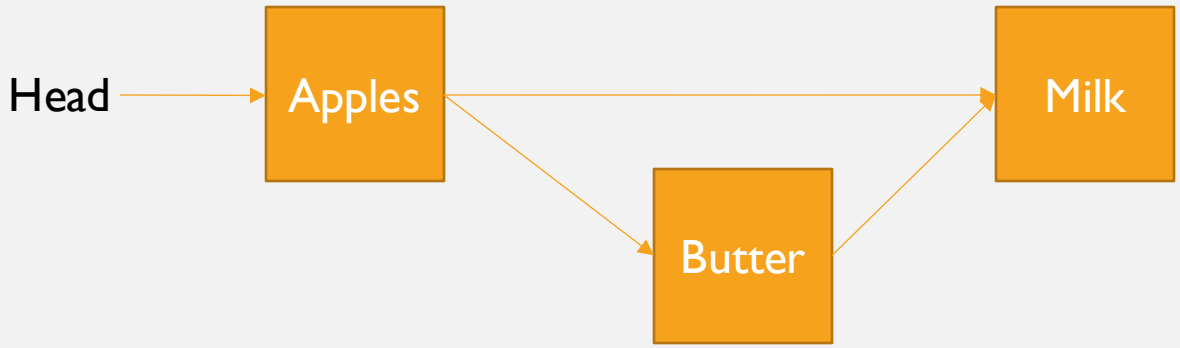

Exercise: Code a function to append an element at the end of a linked list.

# INSERT



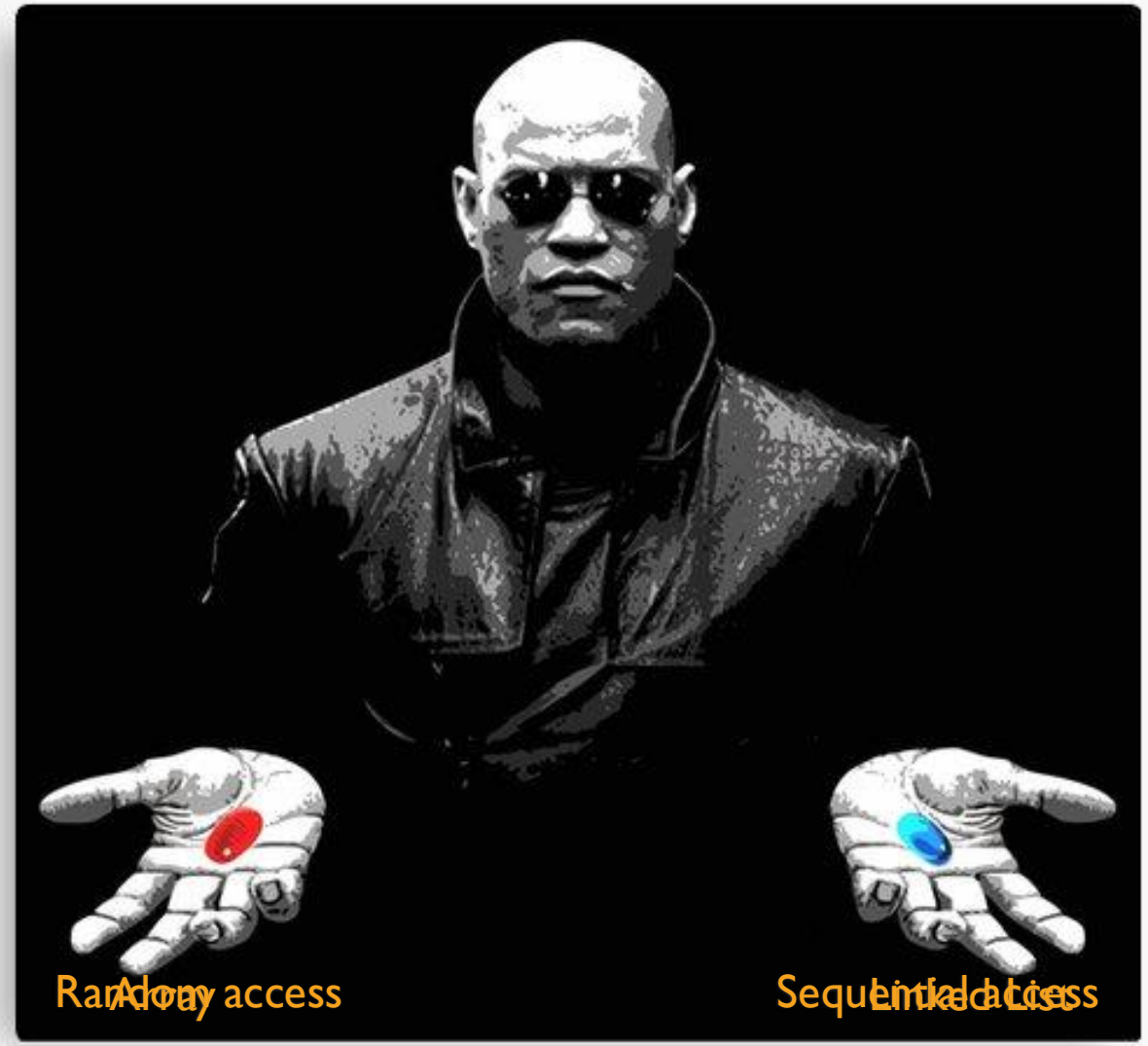

Exercise: Code a function to insert elements in a list, but making sure the list always remains sorted.

# DELETE




Exercise: Code a function to search for an element in a linked list and delete it.


WHEN YOU FAIL TO
BUILD YOUR LINKED LIST
imgflip.com


Random access
Array
Sequential access
Linked List

# REVIEW

# APPENDING AN ITEM TO A LINKED LIST TAKES

- a) O(1)
- b) O(N)
- c) $O(N^2)$

# PREPENDING AN ITEM TO A LINKED LIST TAKES

- a) O(1)
- b) O(N)
- c) $O(N^2)$

# WHAT ABOUT DELETION?

- Main array and linked list operations:

| | ARRAYS | LISTS |
|---|---|---|
| READING | O(1) | O(n) |
| INSERTION | O(n) | O(1) |
| DELETION | O(n) | O(1) |

# EXERCISE

- Suppose you are building an app to keep track of your finances.
  Every day you write down everything you spent money on.
  At the end of the month you review your expenses and sum up how much you spent.


Hint: You have lots of inserts and a few reads.

ARRAY

LINKED LIST

# EXERCISE

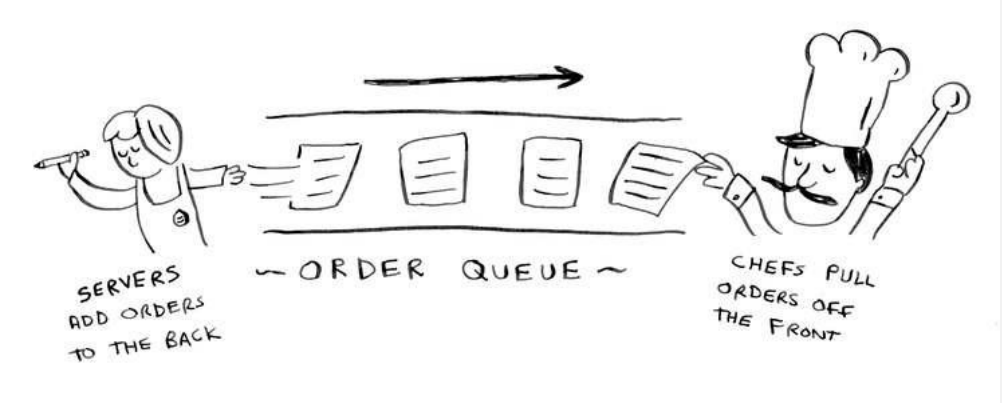
- Now you are tasked to build a TO-DO app. You must keep track of a list where you keep adding tasks, and also mark these already done as "completed".


ARRAY

LINKED LIST

# EXERCISE

- Suppose you are building an app for restaurants to take customer orders. Your app needs to store a linear collection of orders.
  Servers keep adding orders to this list, and chefs take them off the list and make them.

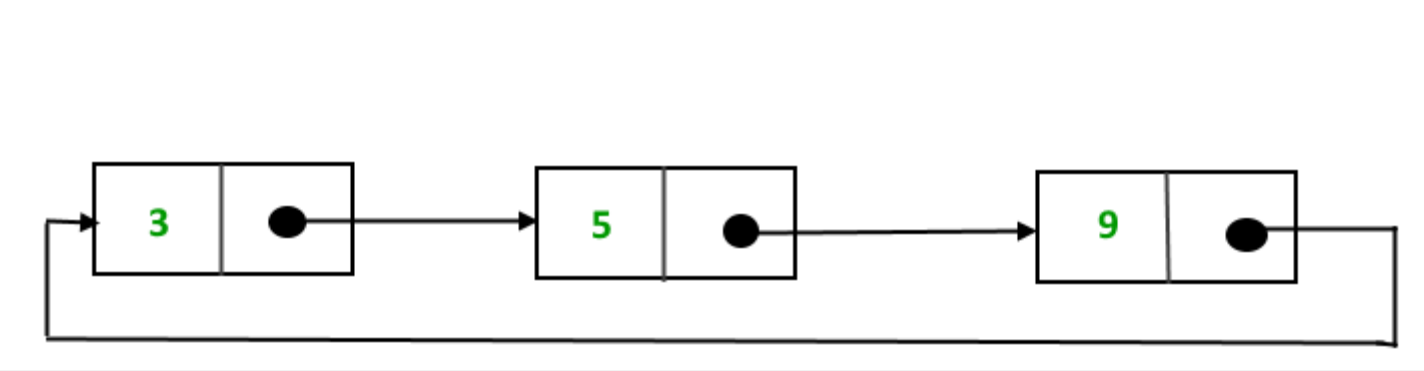



ARRAY

LINKED LIST

# EXERCISE

- Create a function that verifies if a linked list is circular.



# EXERCISE

- Code a function that prints the elements of a singly linked list in reverse order.