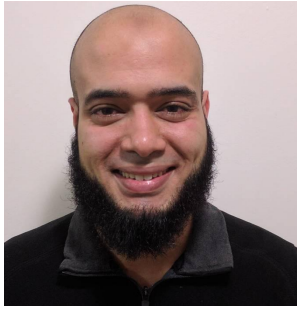
Data Structures

SLL Homework 3

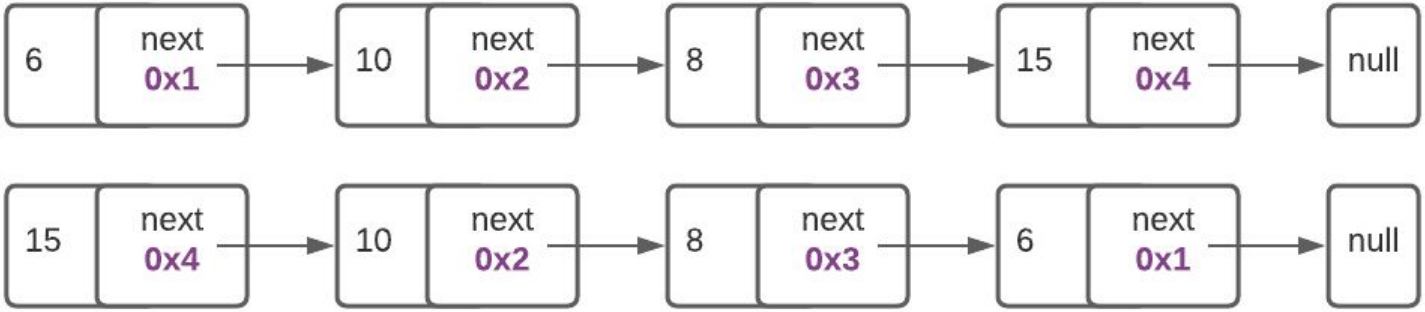
Mostafa S. Ibrahim
Teaching, Training and Coaching for more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



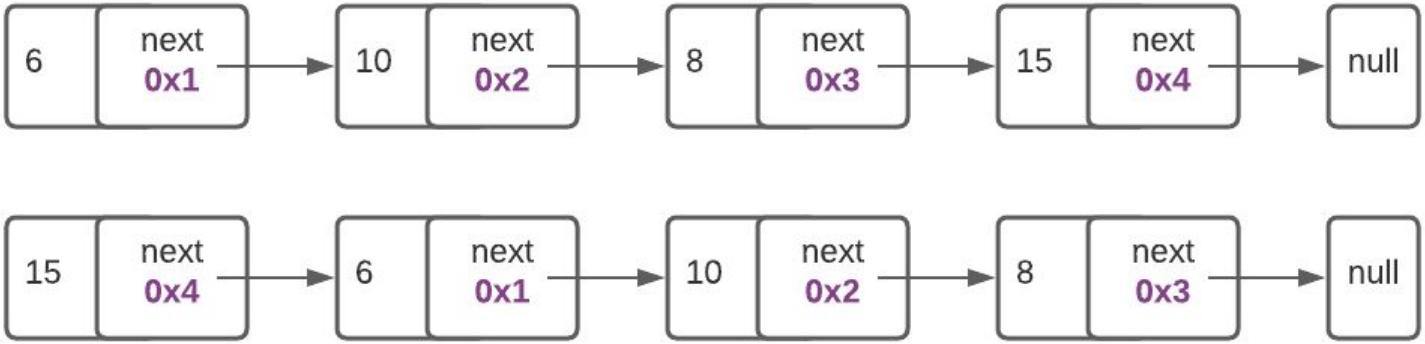
Problem #1: Swap head and tail

- Given a list, we would like to swap the head node with the tail node
  - This swaps the nodes (**addresses**) themselves, not the just the values
  - Look at the nodes before and after.. Observe the addresses
- Tip: draw you procedure step-by-step. This will save a lot of your time
- After the swap, make sure you print out both the values and addresses
- **def** swap_head_tail(self):



Problem #2: Left Rotate

- Given a list, we would like to rotate it to the left k steps (k up to 200000000)
  - A single rotation, takes the first element and send it to the end of the list
- def left_rotate(self, k): Your code should be O(n) time
- The list below is rotated with k=3 (nodes 6 10 8 are shifted back)
  - If k = 1 ⇒ {10, 8, 15, 6}
  - If k = 2 ⇒ {8, 15, 6, 10}
- **def** left_rotate(self, k):



Problem #3: Delete duplicates

- Given an unsorted linked list of numbers, remove any nodes containing repeated numbers - except for the **first node** containing that value
- `def remove_duplicates_from_not_sorted(self):`
- 1, 2, 1, 3, 2, 4, 3, 5, 2 ⇒ 1, 2, 3, 4, 5
- 1, 2, 3, 4, 5 ⇒ 1, 2, 3, 4, 5
- 1, 1, 1 ⇒ 1

Problem #4: Delete last occurance

- Given a linked list of unsorted numbers and a key, remove the final node for which that key occurs
- **1**, 2, 3 - key = 1 ⇒ 2, 3
- 1, 2, 3, 4, **1** - key = 1 ⇒ 1, 2, 3, 4
- 1, 2, 3, **1**, 4 - key = 1 ⇒ 1, 2, 3, 4
- 1, 2, 3, 4 - key = 7 ⇒ 1, 2, 3, 4
- **def** delete_last_occurrence_target(self, target):

Problem #5: Move to the back!

- Given an unsorted list of numbers and a key, move all nodes for which this key matches to the end of the list (nodes order doesn't matter)
- 1, 2, 3, 2, 4, 1 - key = 1 ⇒ 2 3 2 4 1 1
- 1, 2, 3, 1, 2, 4, 1, 7, 1, 8, 1, 1 - key = 1 ⇒ 2 3 2 4 7 8 1 1 1 1 1 1
- Don't create any new nodes
- `def move_key_occurance_to_end(self, target)`

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."