Data Structures

DLL Insertion

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

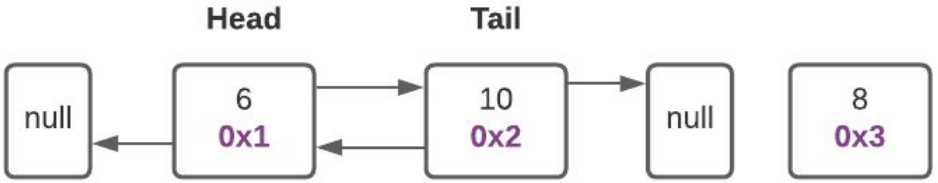
Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Insert end

```
void insert_end(int value) {
    Node* item = new Node(value);
    add_node(item);

    if (!head)
        head = tail = item;
    else {
        link(tail, item);
        tail = item;
    }
    debug_verify_data_integrity();
}
```

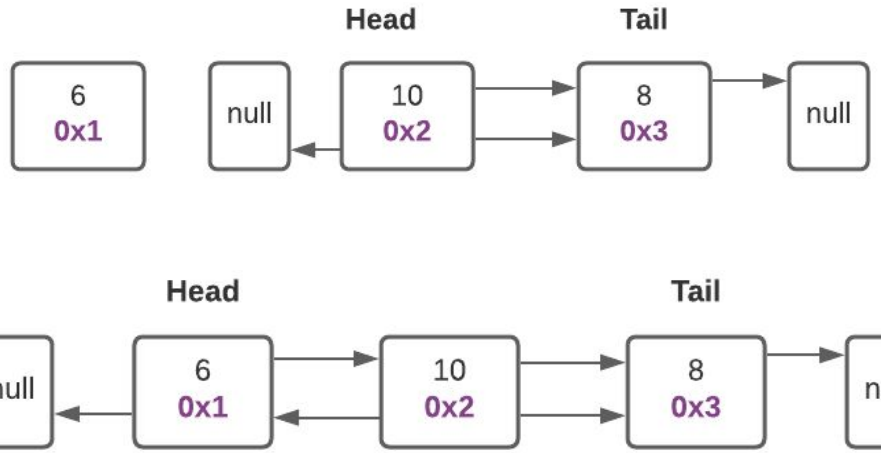
```
void link(Node* first, Node*second) {
    if(first)
        first->next = second;
    if(second)
        second->prev = first;
}
```



Insert front

```
void insert_front(int value) {
    Node* item = new Node(value);
    add_node(item);

    if (!head)
        head = tail = item;
    else {
        link(item, head);
        head = item;
    }
    debug_verify_data_integrity();
}
```



Insert Sorted

- Same logic as homework problem!

```
void insert_sorted(int value) {
    if (!length || value <= head->data)
        insert_front(value);
    else if (tail->data <= value)
        insert_end(value);
    else {
        // Find the node I am less than. Then I am before it
        for (Node *cur = head; cur; cur = cur->next) {
            if (value <= cur->data) {
                embed_after(cur->prev, value);
                break;
            }
        }
    }
```

Insert Sorted

- Given 3 nodes: before, middle and after, we can easily link them!
  - Observe: link function makes our life easier (code and logic)

```
void embed_after(Node* node_before, int value) {
    // Add a node with value between node and its next
    Node* middle= new Node(value);
    ++length;
    debug_add_node(middle);

    Node* node_after = node_before->next;
    link(node_before, middle);
    link(middle, node_after);
}
```

Debugging

- The print now changed to show next/prev
- Verify function make sure u can go forward/backward

```
X       <= [1]  =>      2       head
1       <= [2]  =>      3
2       <= [3]  =>      4
3       <= [4]  =>      5
4       <= [5]  =>      6
5       <= [6]  =>      X       tail
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."