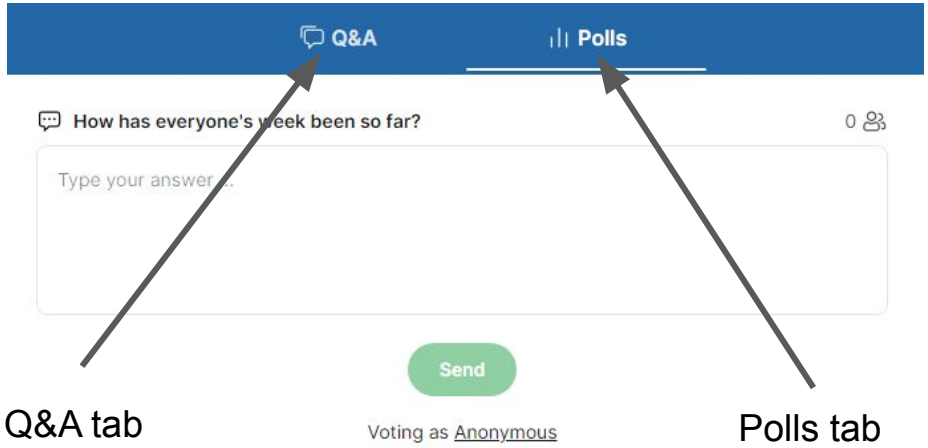
Linked Lists
Knowledge Graphs

Aidan San



Connect to the sli.do at:

go to: **sli.do**

Event code: **3497821**

Feel free to download and follow along with the template code and the slides from:

go to: **https://tinyurl.com/as033022**

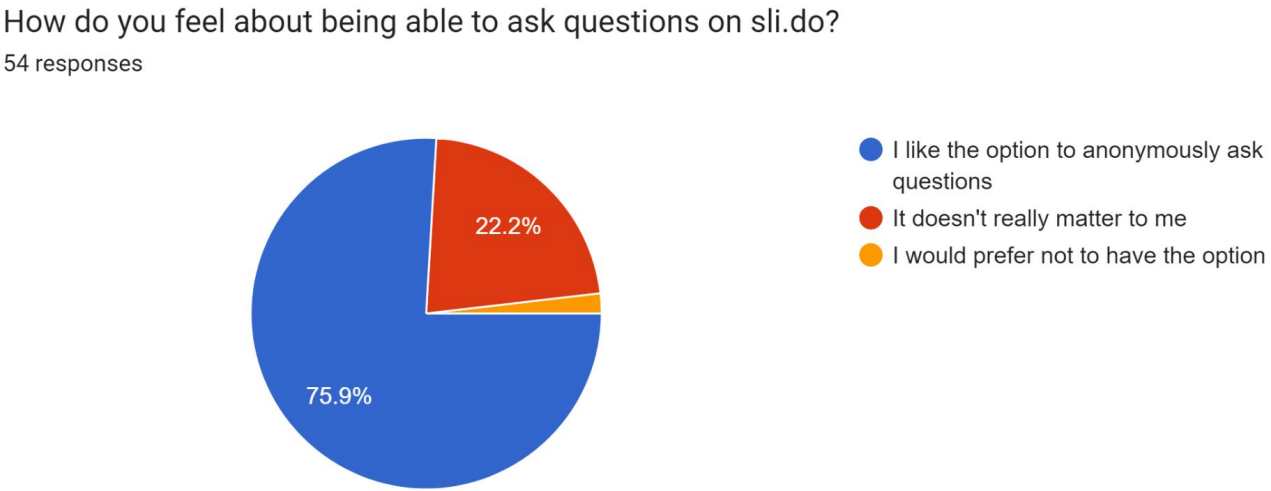
Teaching Philosophy

- Making students feel comfortable in class
- Adapting to students with diverse backgrounds

sli.do: 3497821

Teaching Philosophy

- Making students feel comfortable in class
- Adapting to students with diverse backgrounds
  - Sli.do for anonymous questions/polling



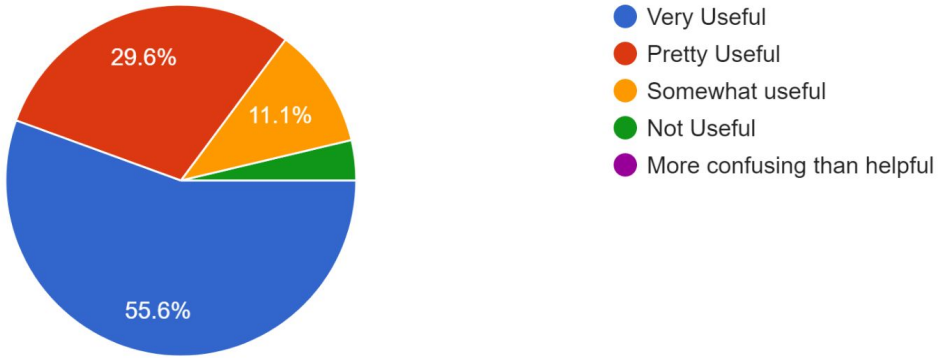
sli.do: 3497821

Teaching Philosophy

- Making students feel comfortable in class
- Adapting to students with diverse backgrounds
  - Sli.do for anonymous questions/polling

How helpful do you find the the instructor's asking "test" questions on sli.do?

54 responses



sli.do: 3497821

Teaching Philosophy

- Making students feel comfortable in class
- Adapting to students with diverse backgrounds
  - Sli.do for anonymous questions/polling
  - Offer Zoom lectures
  - Record lectures
  - Optional Attendance
  - Have slides available ahead of time

sli.do: 3497821

Context - Linked Lists

- Part of a data structures course (CS 2100)
- Students have already learned about
  - regular lists
  - big O notation
  - classes
- I would normally open up the class to ask questions from previous lectures

sli.do: 3497821

Linked Lists

Aidan San

sli.do: 3497821

Overview

1. Why Linked Lists?
2. What is a Linked List?
3. How to add things to the start of the list?
4. How to remove things from the front of the list?
5. How do we iterate through the list?
6. Live coding
7. Big O

sli.do: 3497821

Let’s say you’re working at Bodo’s

- **Task:** Store a queue of orders for the kitchen to prepare

Bodo’s Bagels Orders

1. Cream Cheese
2. Plain
3. Chicken Salad


Cream Cheese


Plain

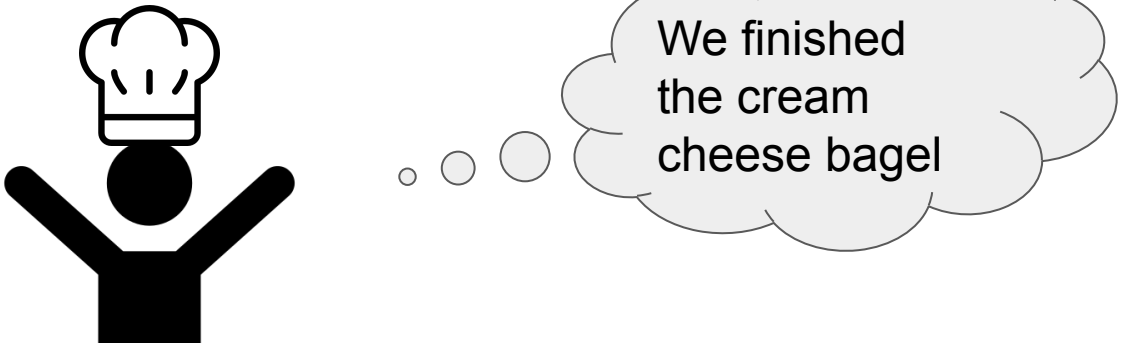

Chicken Salad

sli.do: 3497821

What happens when Bodo's completes an order?

Bodo's Bagels Orders

1. Cream Cheese
2. Plain
3. Chicken Salad



sli.do: 3497821

What happens when Bodo's completes an order?

Bodo's Bagels Orders

1.
2. Plain
3. Chicken Salad

sli.do: 3497821

What happens when Bodo's completes an order?

Bodo's Bagels Orders

1. Plain
2. Chicken Salad

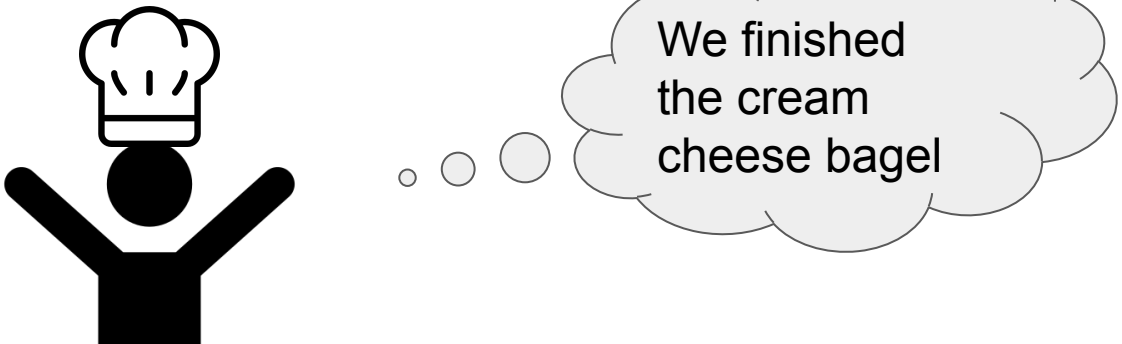
sli.do: 3497821

What happens when Bodo's completes an order?

<u>Bodo's Bagels Orders</u>

1. Plain
2. Chicken Salad

We remove the items from the beginning!



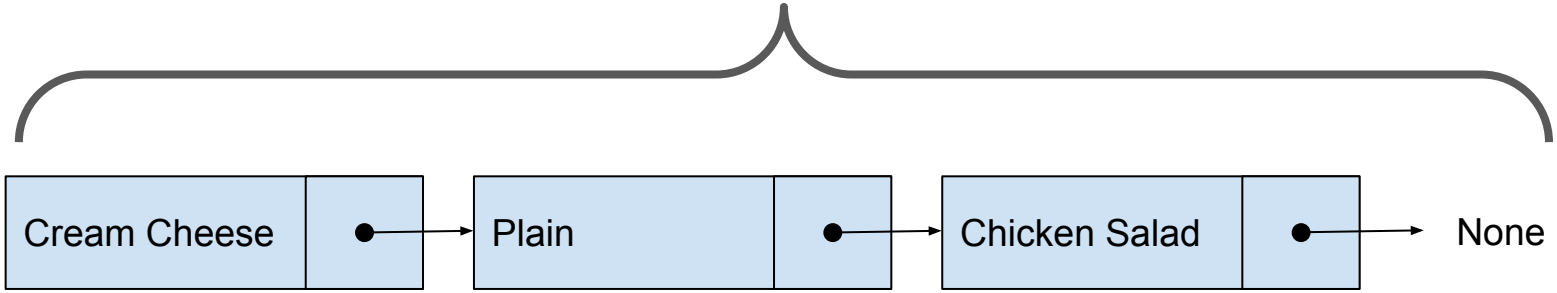
sli.do: 3497821

Motivation

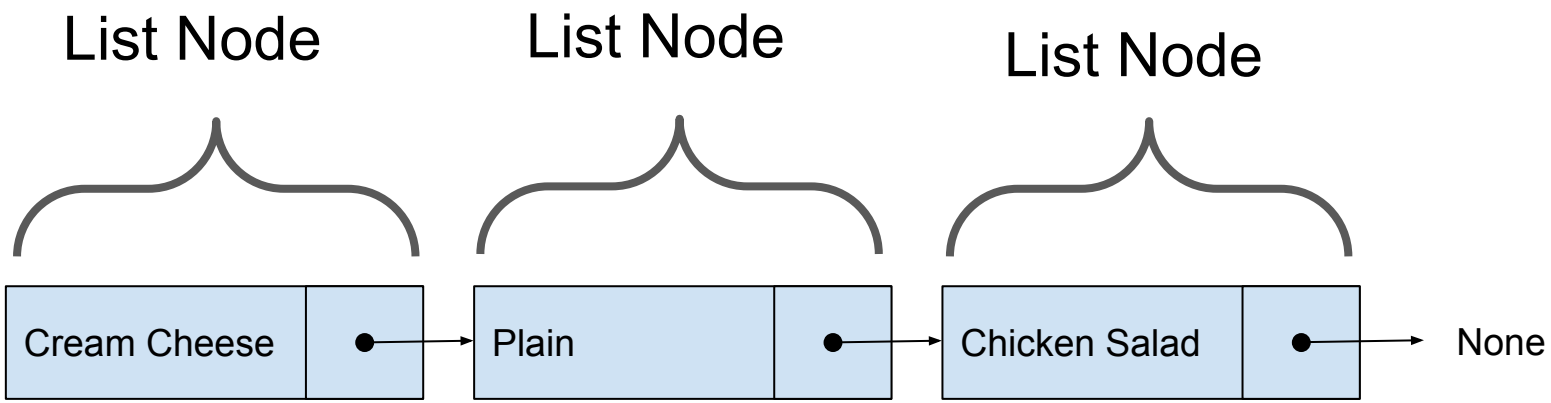
- We want to remove from the front of the list quickly
- Regular lists: removing from front - *O(n)*
- Can we do better?

sli.do: 3497821

Linked List

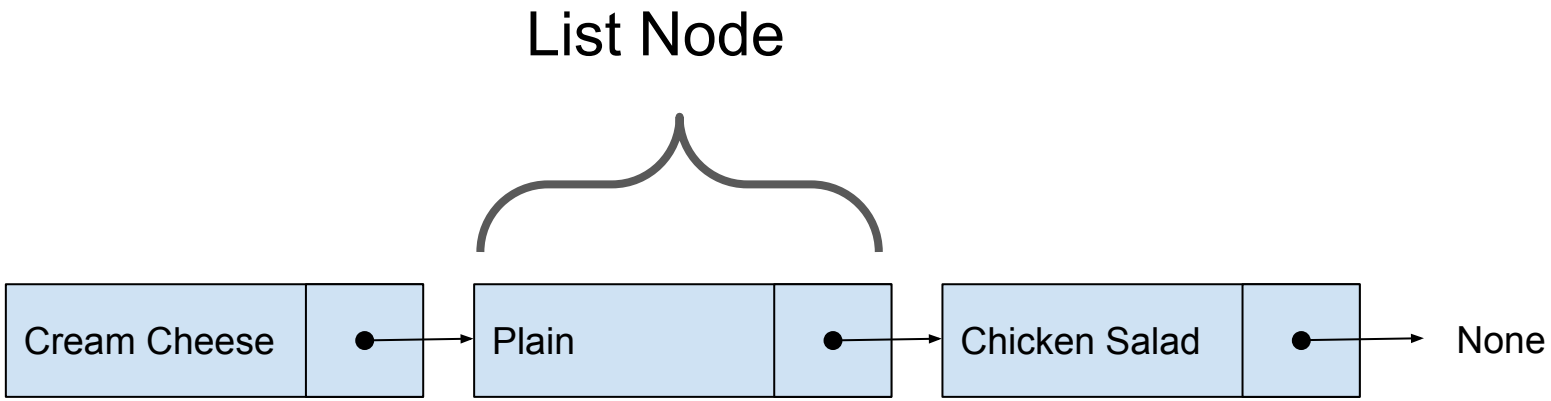


sli.do: 3497821



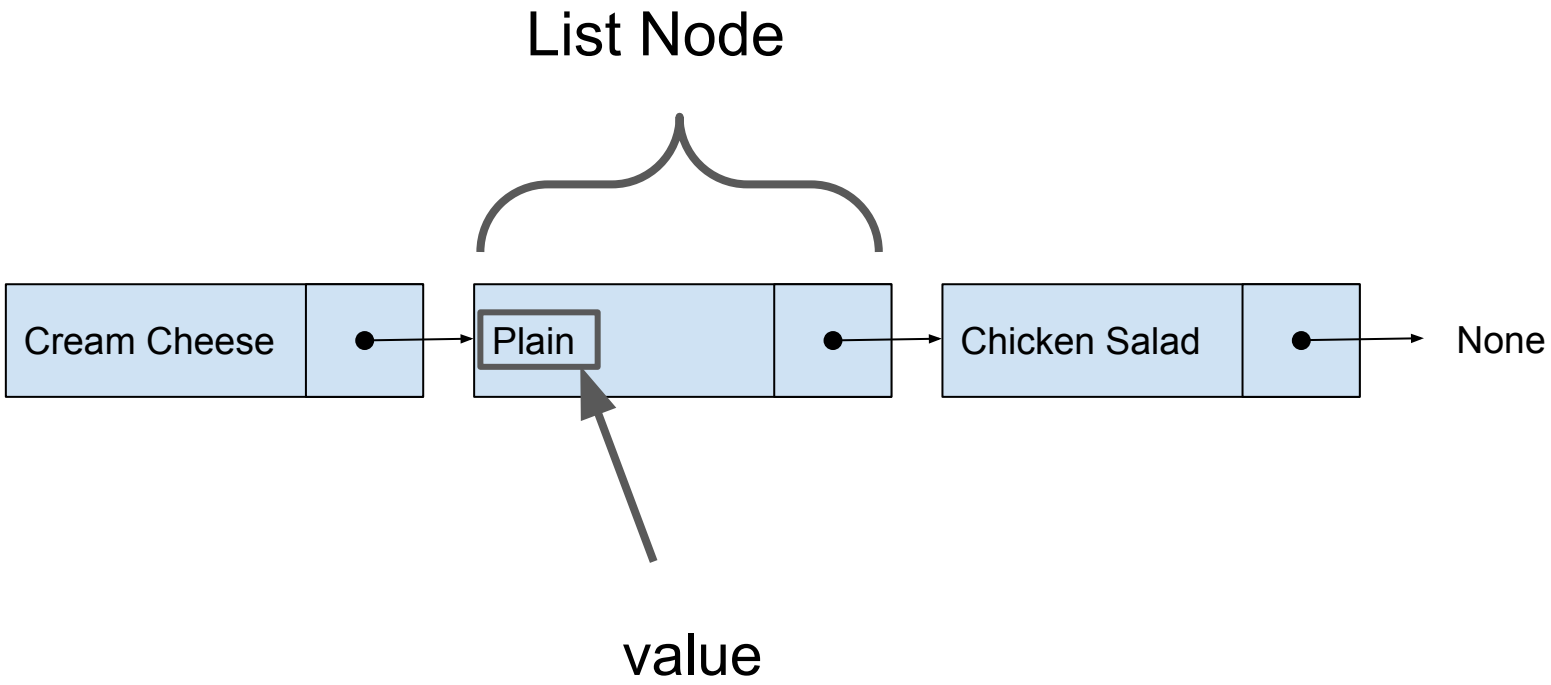
sli.do: 3497821

List Nodes have 2 member variables



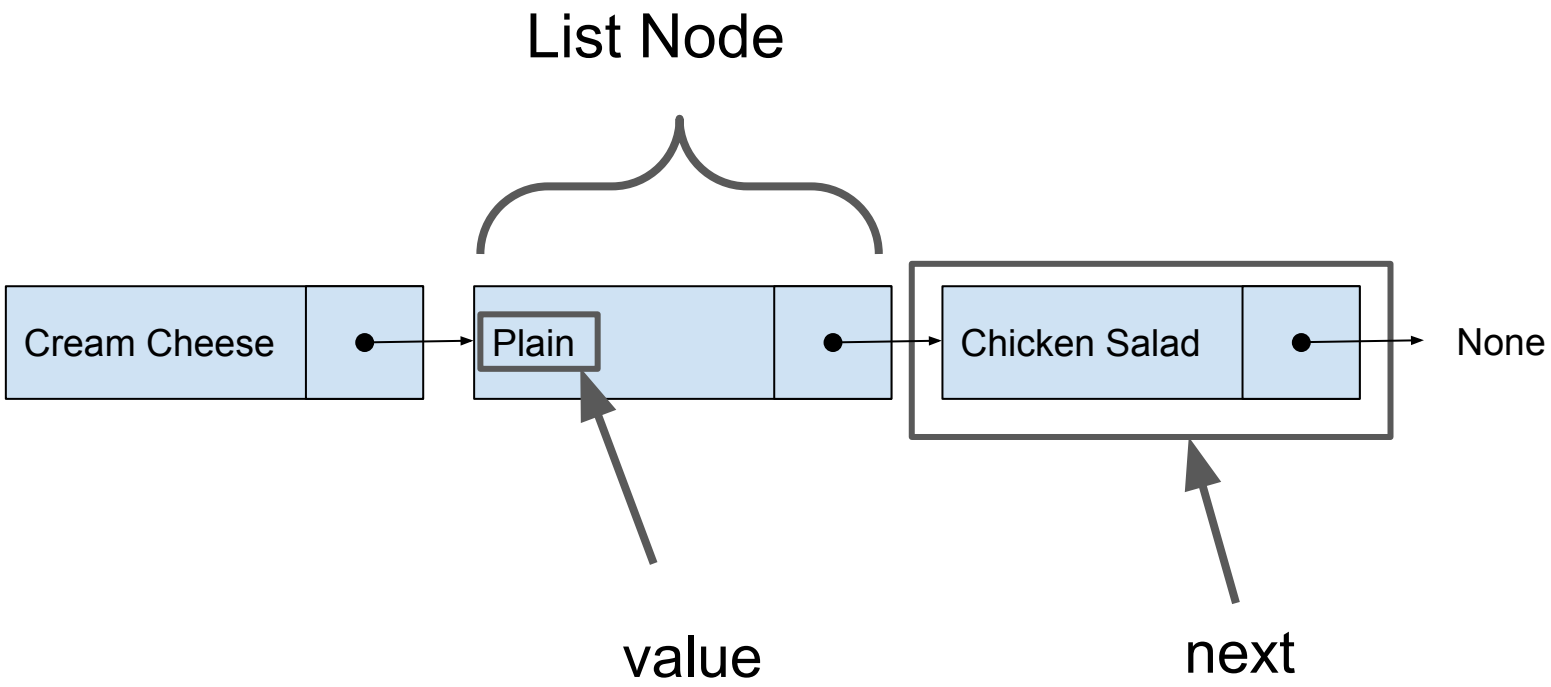
sli.do: 3497821

List Nodes have 2 member variables



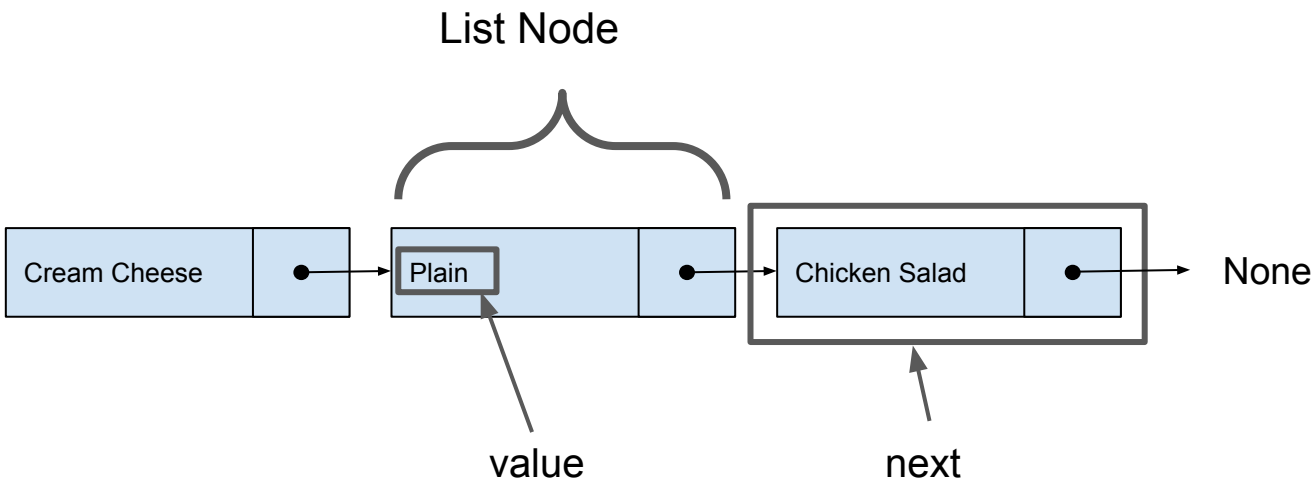
sli.do: 3497821

List Nodes have 2 member variables



sli.do: 3497821

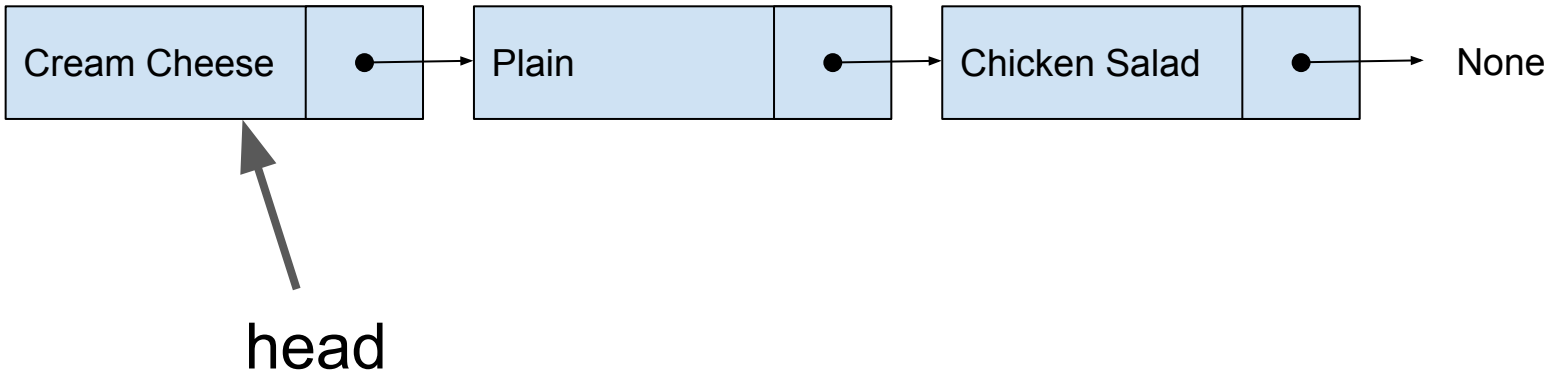
```
class ListNode:
    def __init__(self, val):
        self.val = val
        self.next = None
```



sli.do: 3497821

Linked Lists have a “head”

The first List Node in a Linked List



sli.do: 3497821

What if a VIP shows up and Bodo’s needs to rush an order?

sli.do: 3497821

What if a VIP shows up and Bodo’s needs to rush an order?

sli.do: 3497821

What if a VIP shows up and Bodo's needs to rush an order?

I want a ham + egg, and am late for my concert!

sli.do: 3497821

Bodo’s adds an order to the front

Bodo’s Bagels Orders

1. Cream Cheese
2. Plain
3. Chicken Salad

I want a ham + egg, and am late for my concert!

sli.do: 3497821

Bodo’s adds an order to the front

Bodo’s Bagels Orders

1.
2. Cream Cheese
3. Plain
4. Chicken Salad

I want a ham + egg, and am late for my concert!

sli.do: 3497821

Bodo’s adds an order to the front

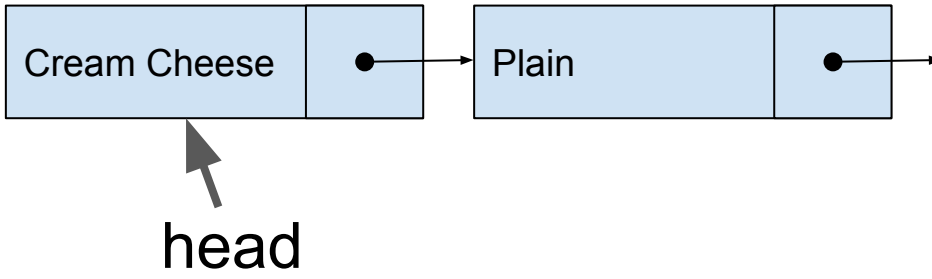
Bodo’s Bagels Orders

1. Ham + Egg
2. Cream Cheese
3. Plain
4. Chicken Salad

I want a ham + egg, and am late for my concert!

sli.do: 3497821

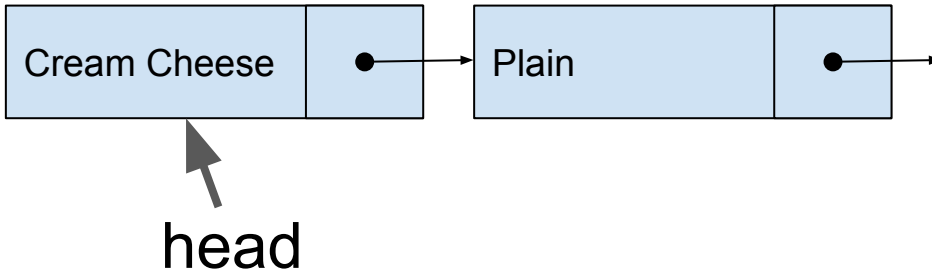
How do we add a new node to the front?



sli.do: 3497821

How do we add a new node to the front?

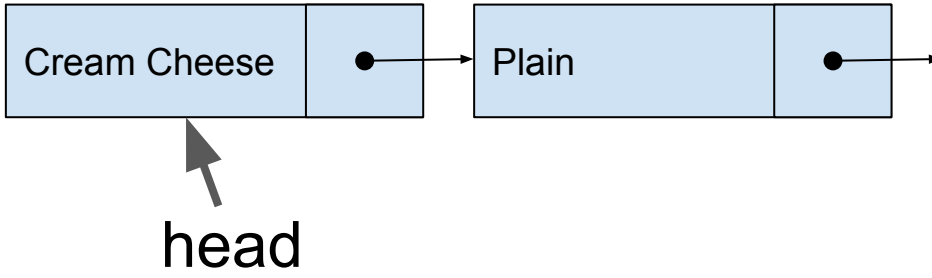
1. Make a new node



sli.do: 3497821

How do we add a new node to the front?

1. Make a new node

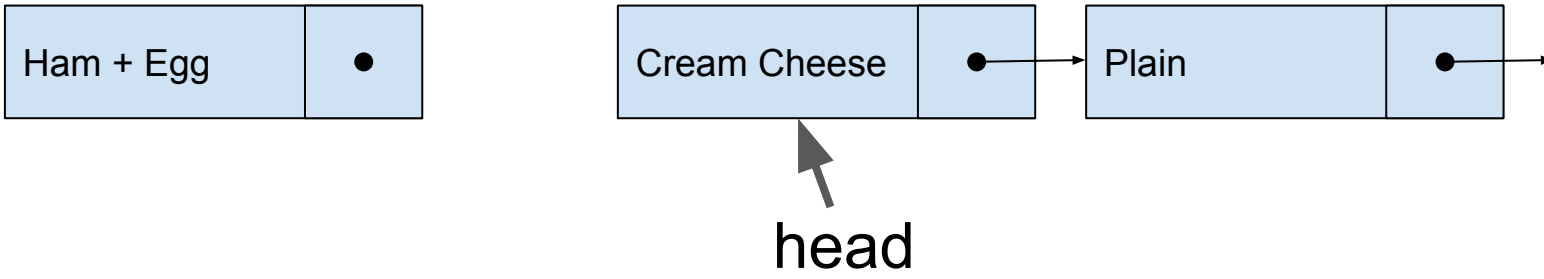



```
new_node = ListNode("Ham + Egg")
```

sli.do: 3497821

How do we add a new node to the front?

1. Make a new node

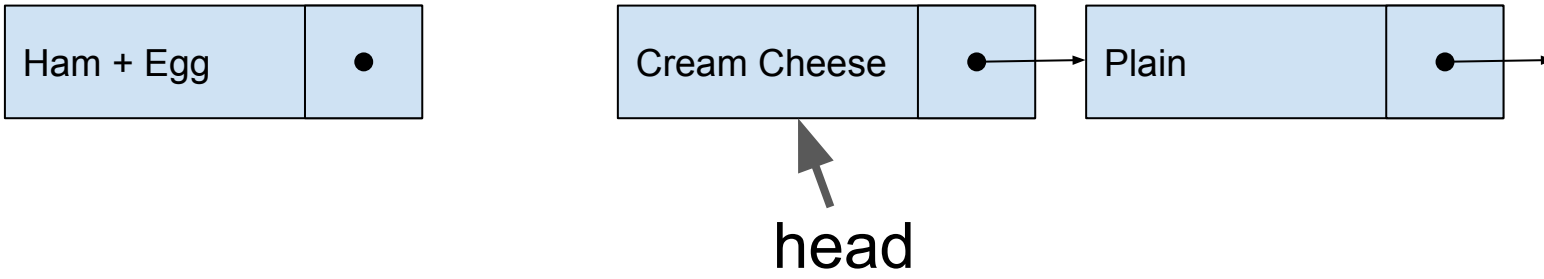



```
new_node = ListNode("Ham + Egg")
```

sli.do: 3497821

How do we add a new node to the front?

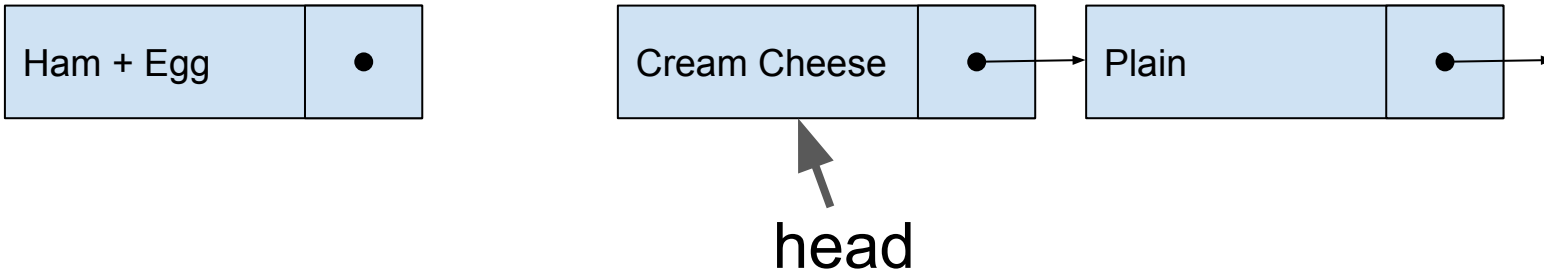
1. Make a new node
2. Connect the new node to the current list (update its “next”)



sli.do: 3497821

How do we add a new node to the front?

1. Make a new node
2. Connect the new node to the current list (update its “next”)

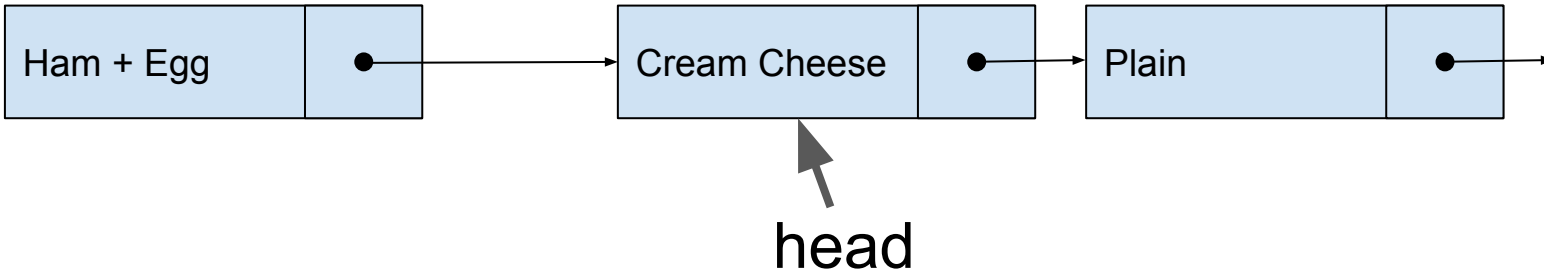



```
new_node.next = head
```

sli.do: 3497821

How do we add a new node to the front?

1. Make a new node
2. Connect the new node to the current list (update its “next”)

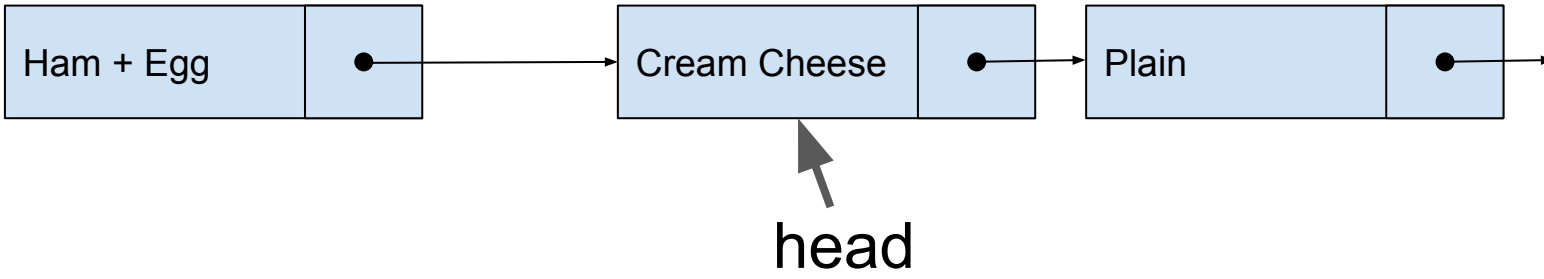



```
new_node.next = head
```

sli.do: 3497821

How do we add a new node to the front?

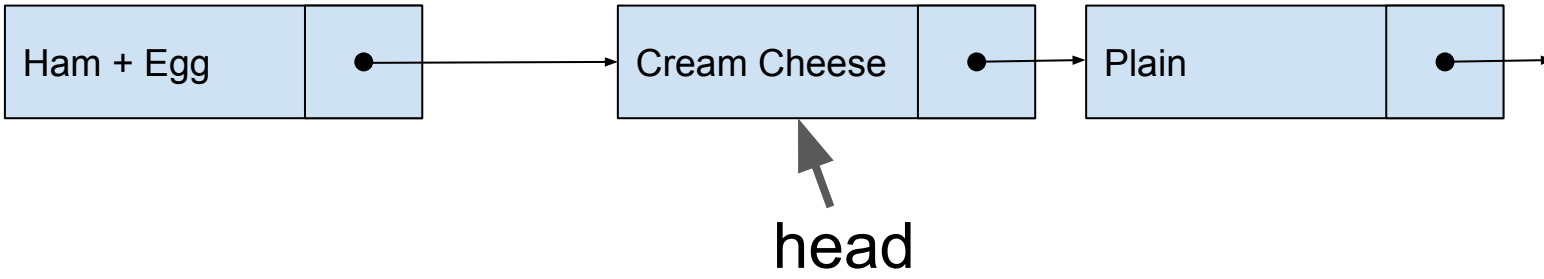
1. Make a new node
2. Connect the new node to the current list (update its “next”)
3. **Sli.do: What’s the next step?**



sli.do: 3497821

How do we add a new node to the front?

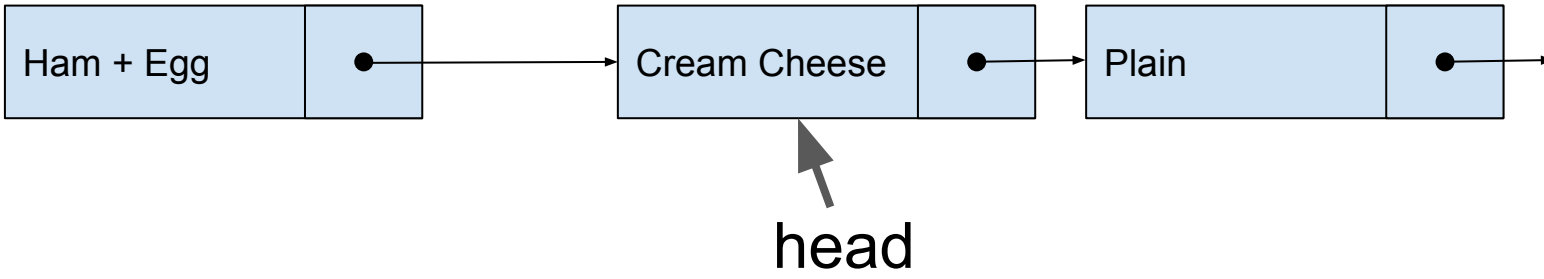
1. Make a new node
2. Connect the new node to the current list (update its “next”)
3. Update “head”



sli.do: 3497821

How do we add a new node to the front?

1. Make a new node
2. Connect the new node to the current list (update its “next”)
3. Update “head”

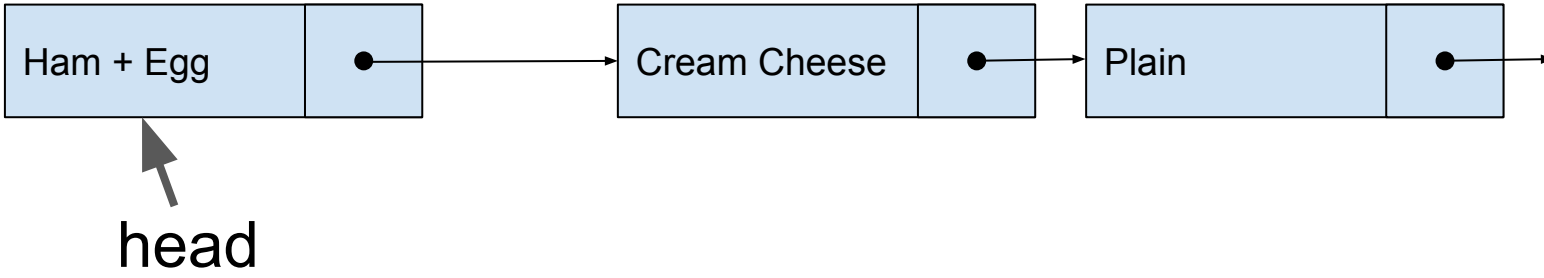



```
head = new_node
```

sli.do: 3497821

How do we add a new node to the front?

1. Make a new node
2. Connect the new node to the current list (update its “next”)
3. Update “head”




```
head = new_node
```

sli.do: 3497821

Live Coding

sli.do: 3497821

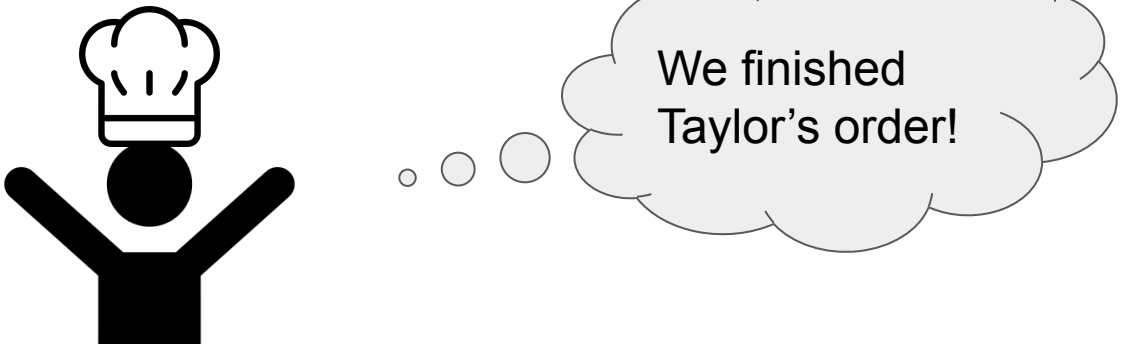
What happens after Bodo’s has finished cooking Taylor’s order?

sli.do: 3497821

Bodo’s removes an order from the front

Bodo’s Bagels Orders

1. Ham + Egg
2. Cream Cheese
3. Plain
4. Chicken Salad

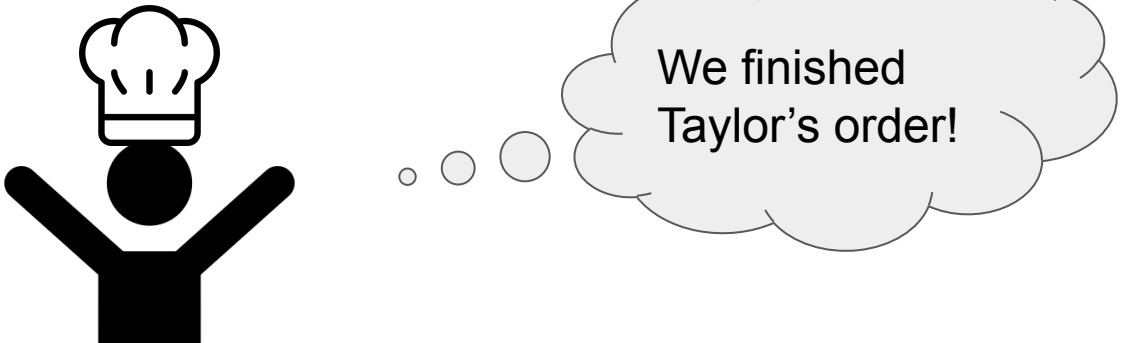


sli.do: 3497821

Bodo’s removes an order from the front

Bodo’s Bagels Orders

1.
2. Cream Cheese
3. Plain
4. Chicken Salad

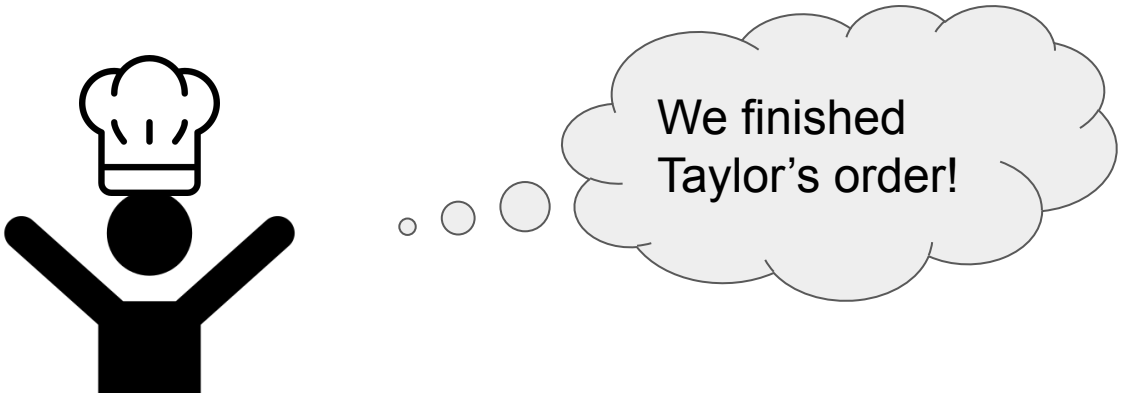


sli.do: 3497821

Bodo’s removes an order from the front

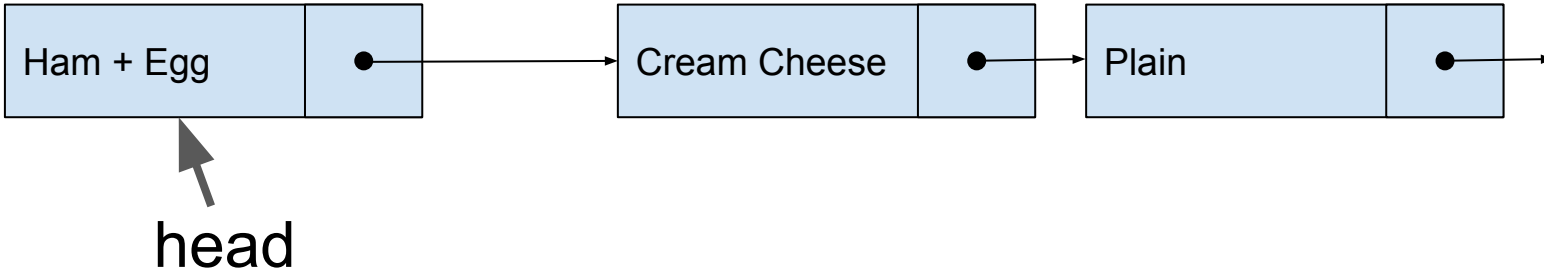
Bodo’s Bagels Orders

1. Cream Cheese
2. Plain
3. Chicken Salad



sli.do: 3497821

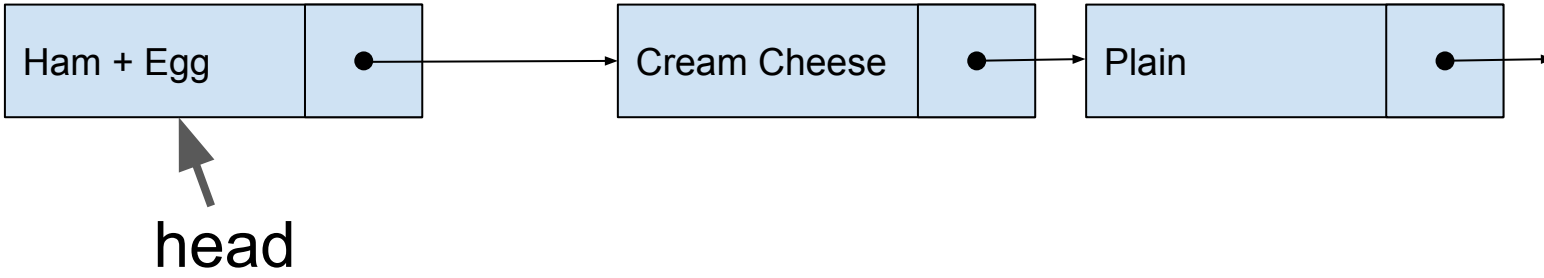
How do we remove a node from the front?



sli.do: 3497821

How do we remove a node from the front?

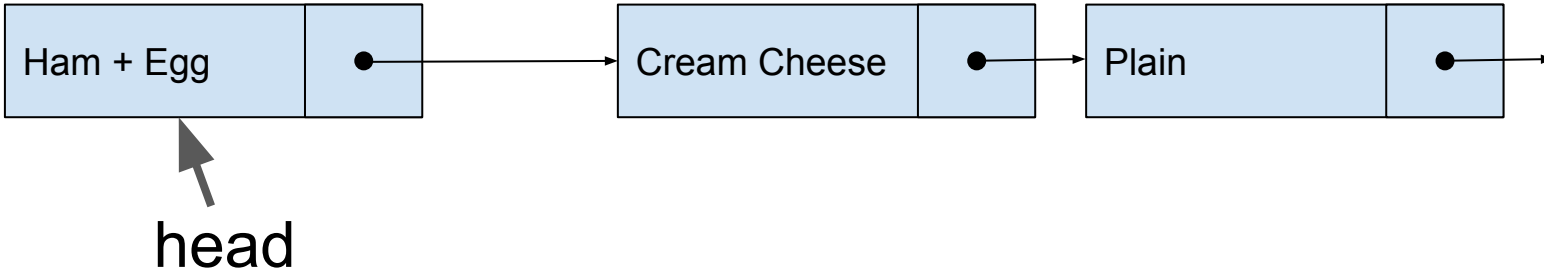
1. Change the head reference



sli.do: 3497821

How do we remove a node from the front?

1. Change the head reference

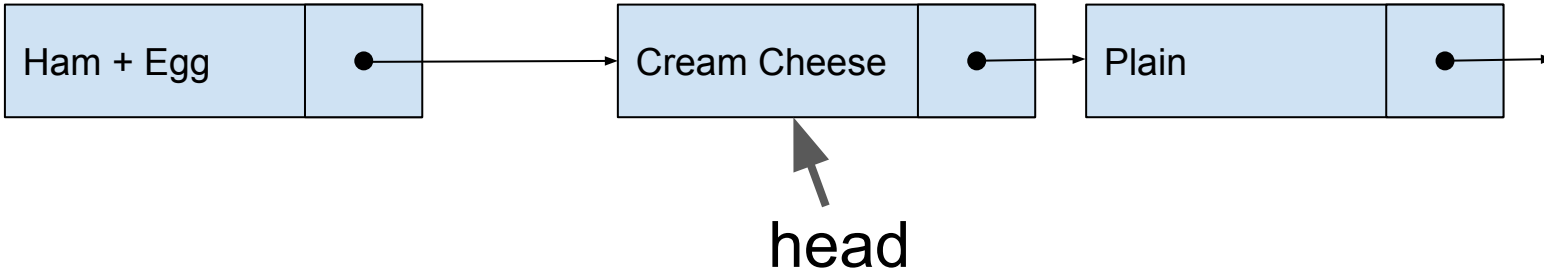



```
head = head.next
```

sli.do: 3497821

How do we remove a node from the front?

1. Change the head reference



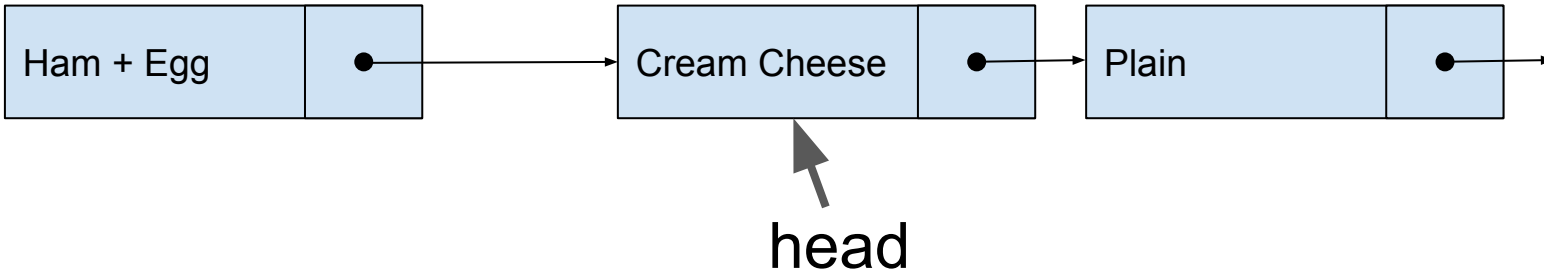

```
head = head.next
```

sli.do: 3497821

How do we remove a node from the front?

1. Change the head reference

Python will garbage collect the previous node for me!



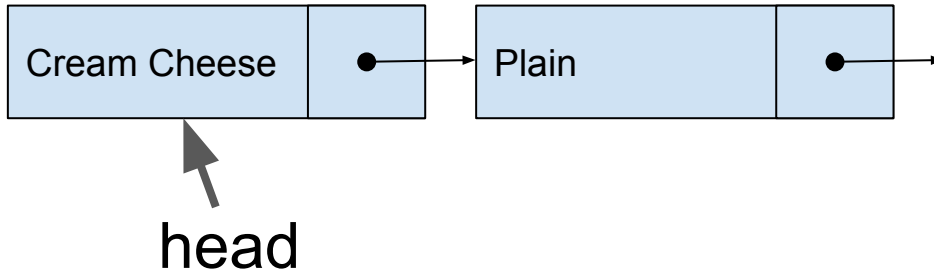

```
head = head.next
```

sli.do: 3497821

How do we remove a node from the front?

1. Change the head reference

Python will garbage collect the previous node for me!



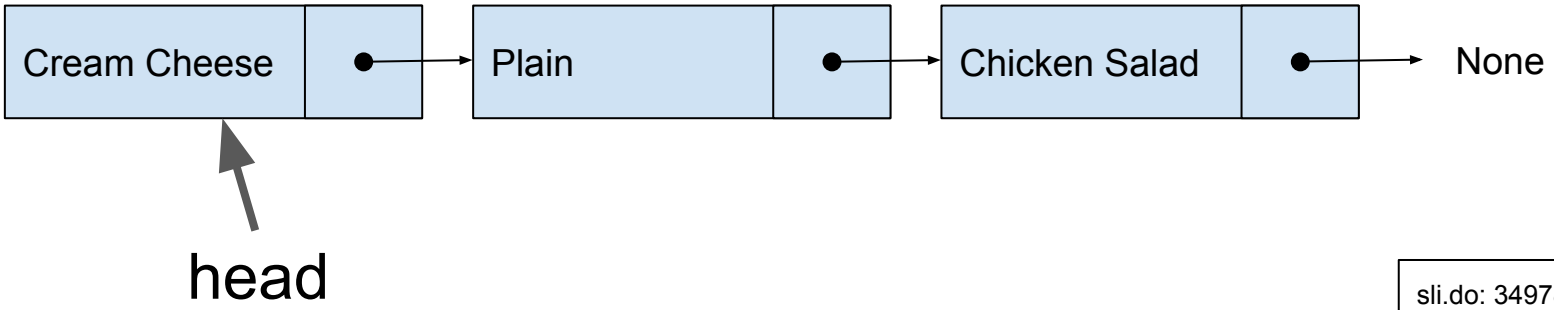

```
head = head.next
```

sli.do: 3497821

What if Bodo’s wants to view all of their current orders?

sli.do: 3497821

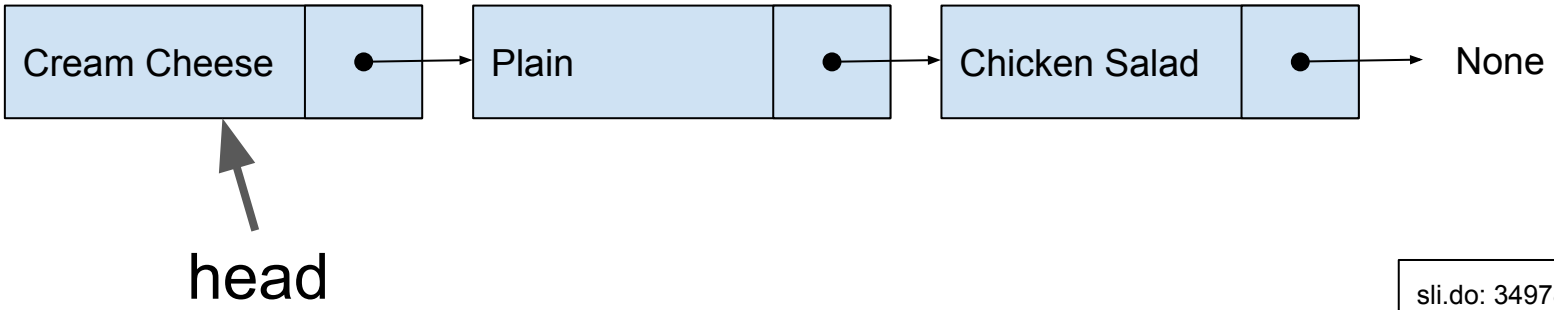
How do we iterate through the Linked List?



sli.do: 3497821

How do we iterate through the Linked List?

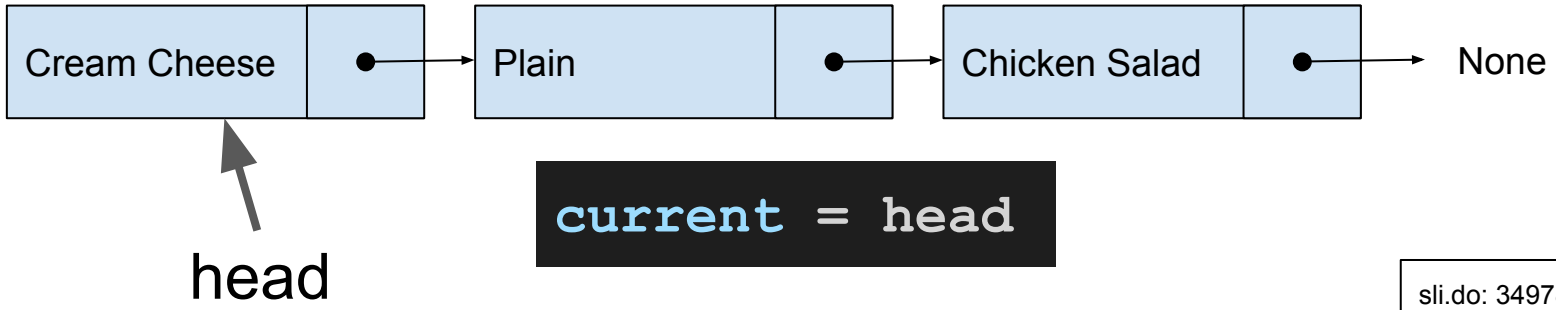
1. Make a new reference (“current”) at the start of the list



sli.do: 3497821

How do we iterate through the Linked List?

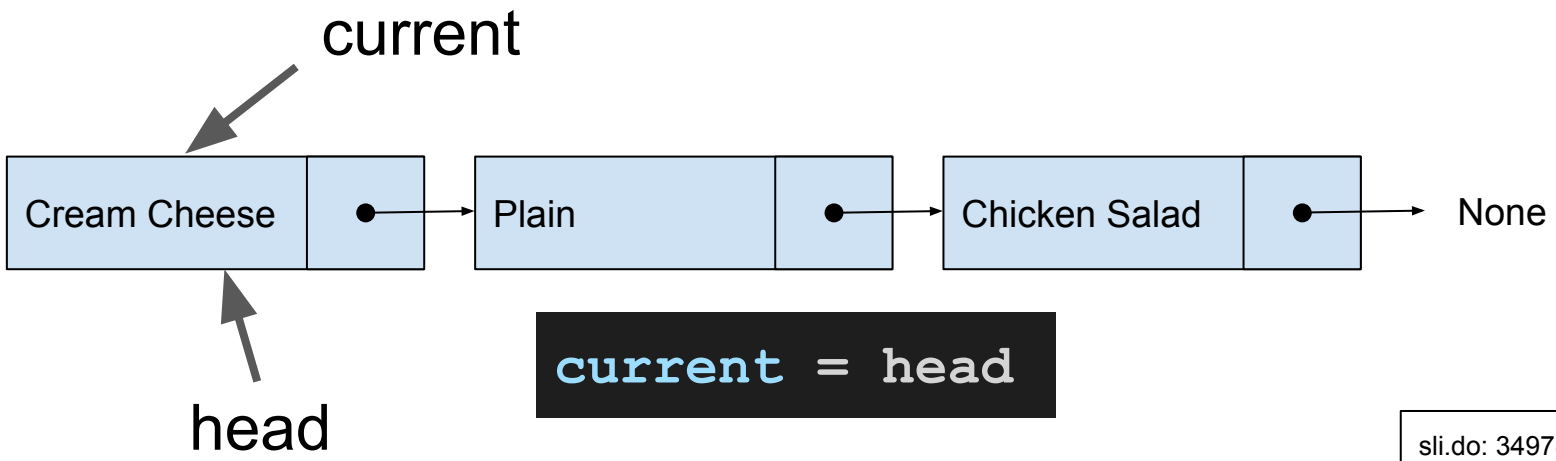
1. Make a new reference (“current”) at the start of the list



sli.do: 3497821

How do we iterate through the Linked List?

1. Make a new reference (“current”) at the start of the list

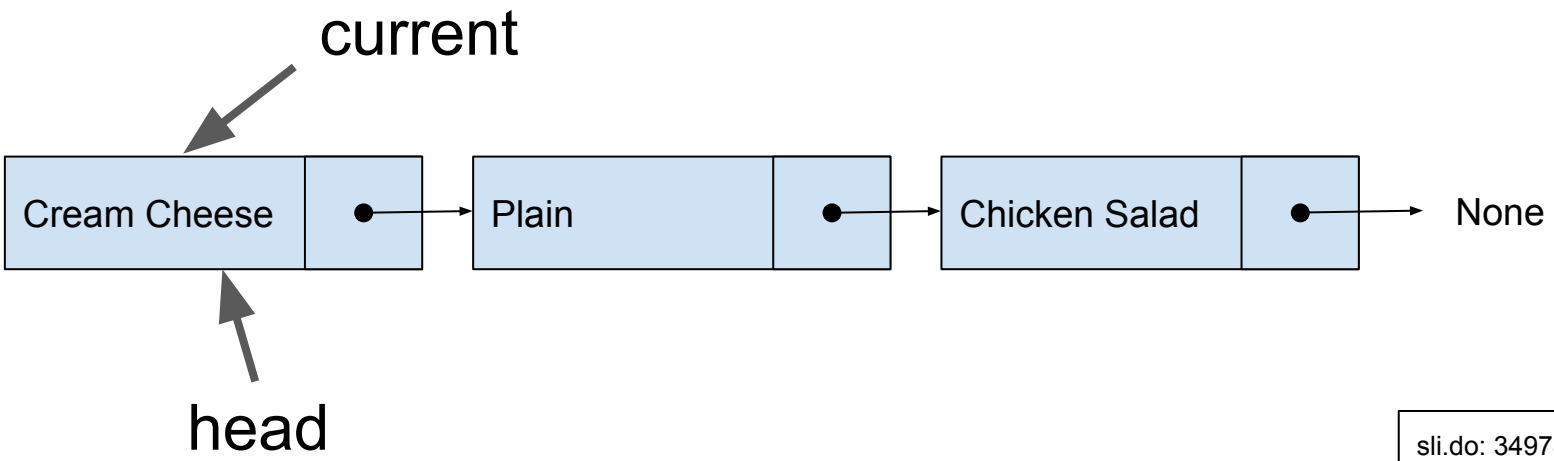



```
current = head
```

sli.do: 3497821

How do we iterate through the Linked List?

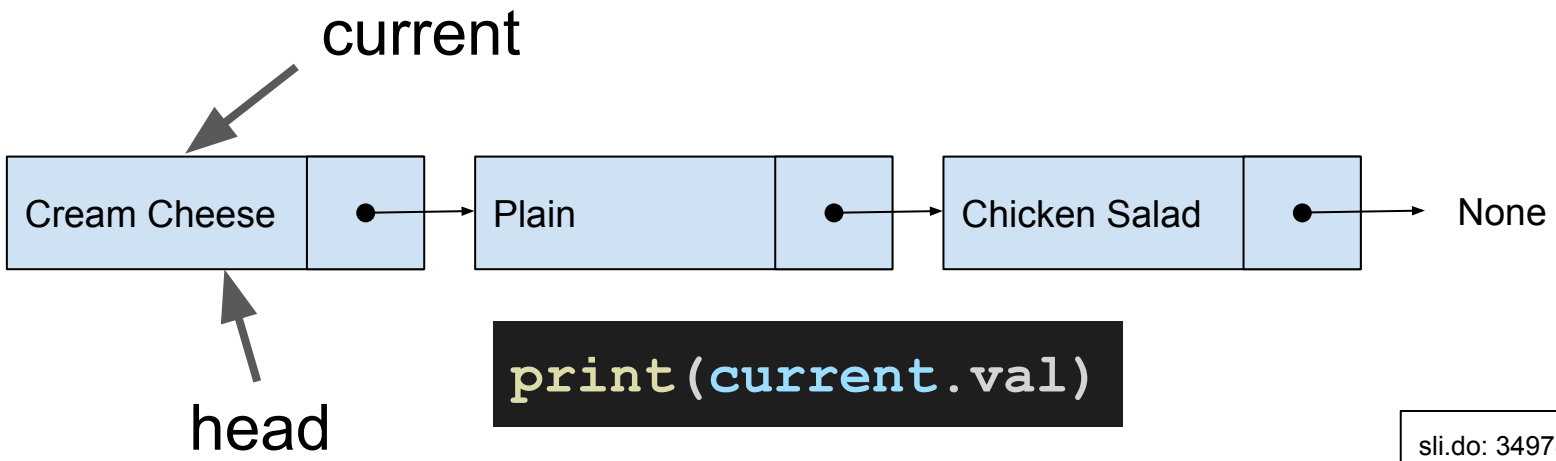
1. Make a new reference (“current”) at the start of the list
2. Print out “current”



sli.do: 3497821

How do we iterate through the Linked List?

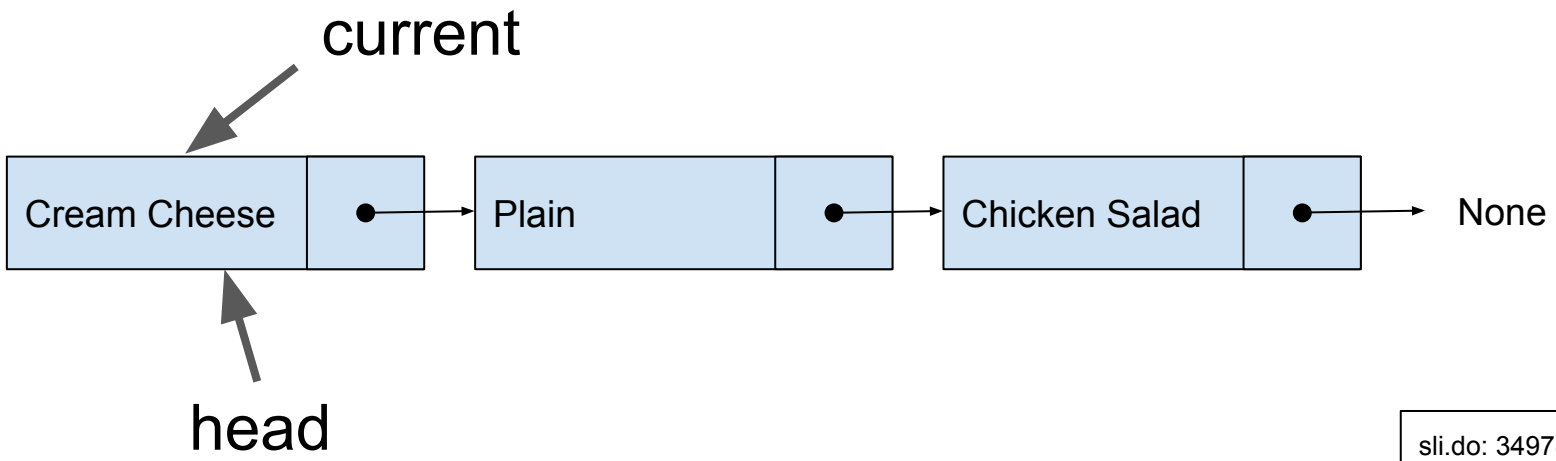
1. Make a new reference (“current”) at the start of the list
2. Print out “current”



sli.do: 3497821

How do we iterate through the Linked List?

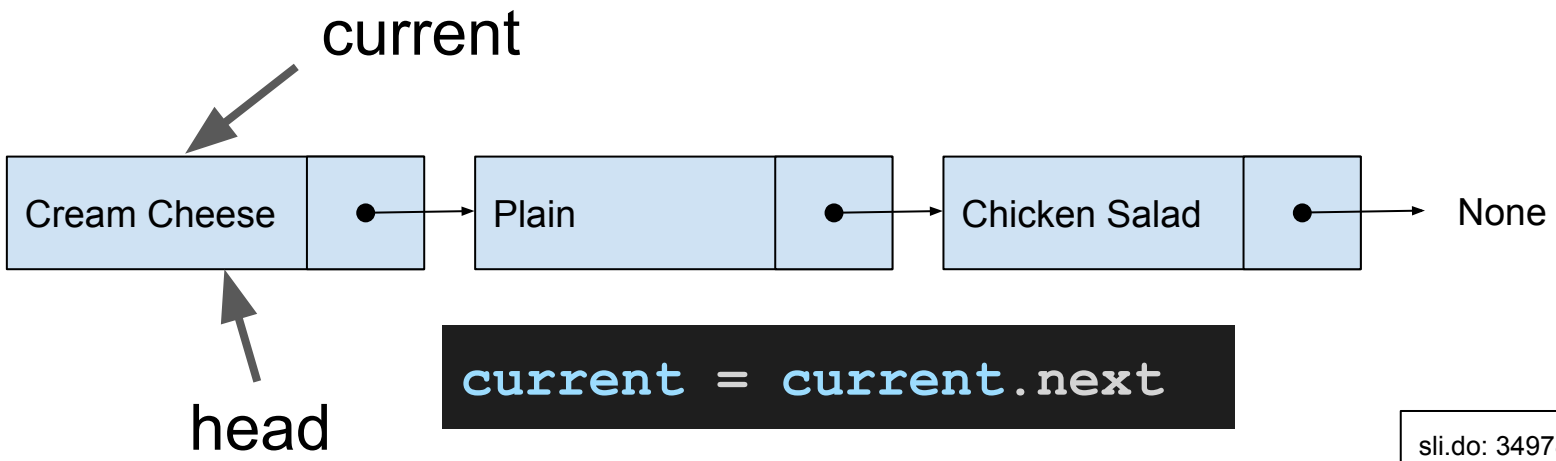
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node



sli.do: 3497821

How do we iterate through the Linked List?

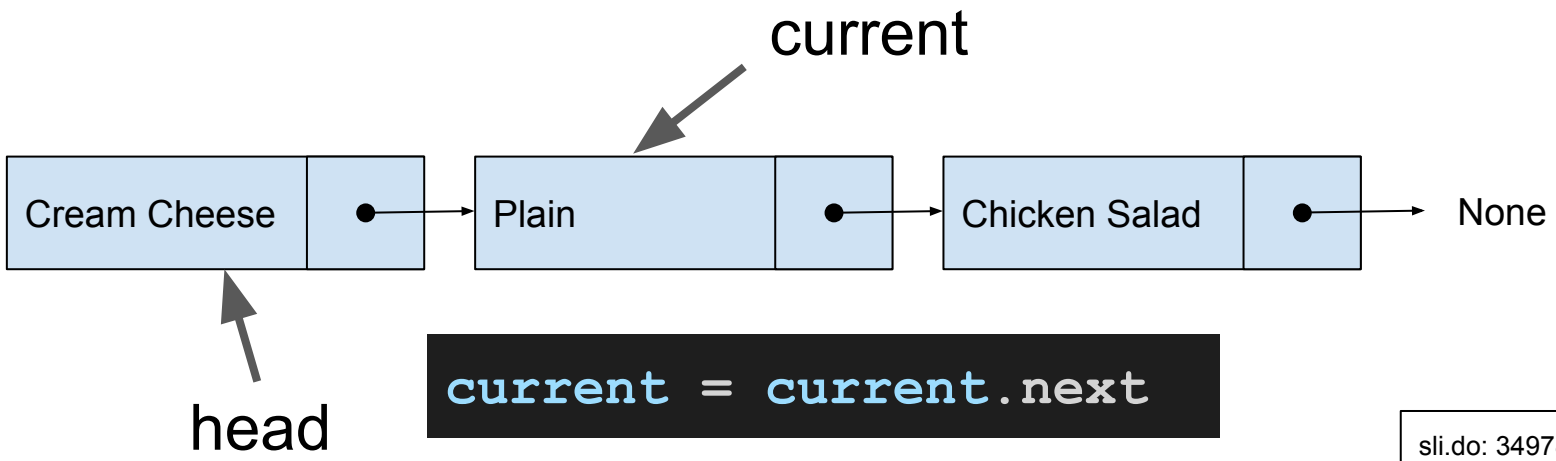
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node



sli.do: 3497821

How do we iterate through the Linked List?

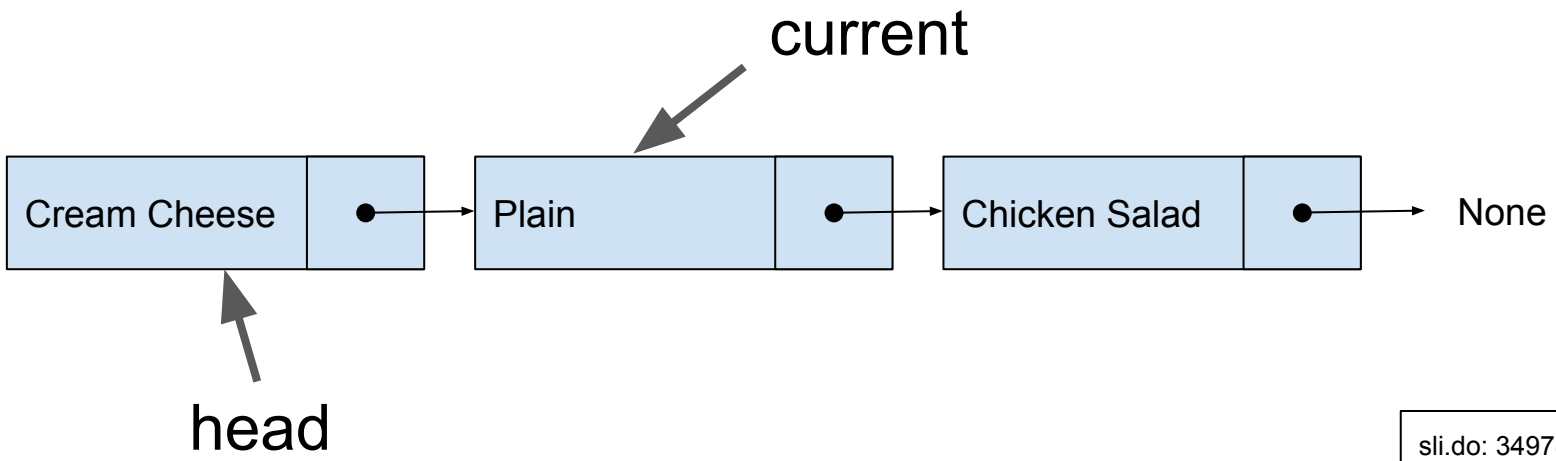
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node



sli.do: 3497821

How do we iterate through the Linked List?

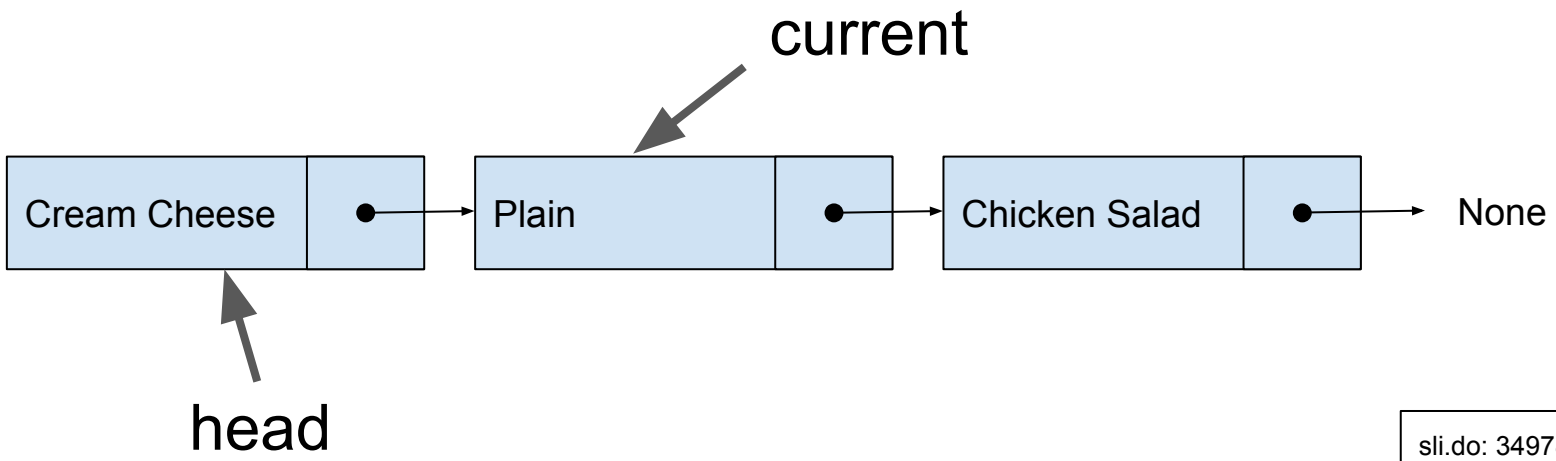
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node



sli.do: 3497821

How do we iterate through the Linked List?

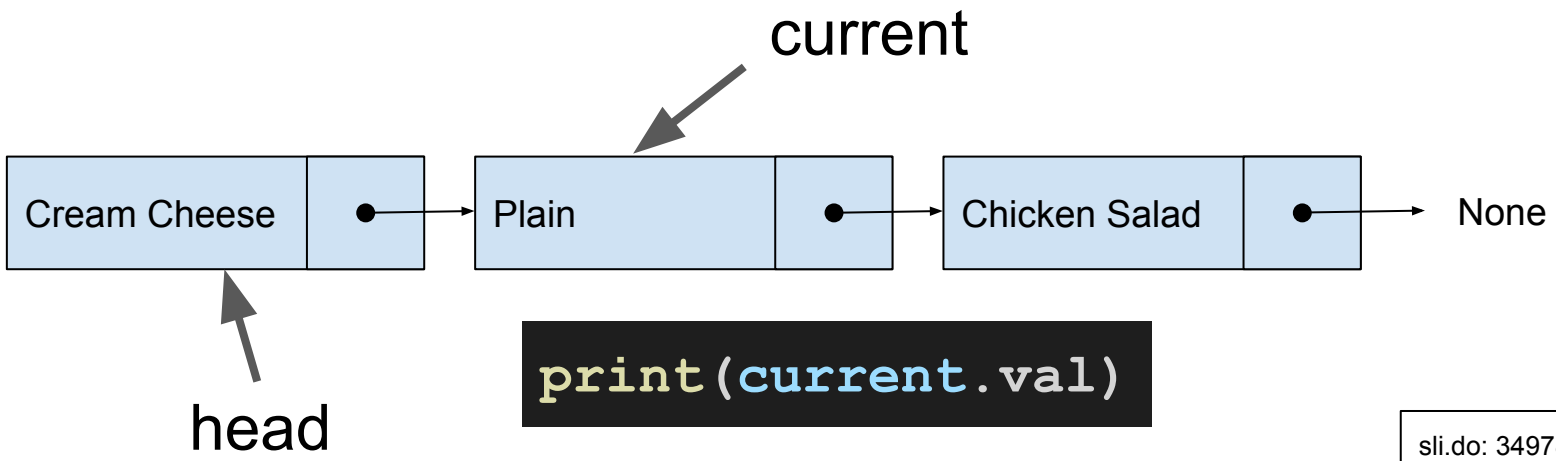
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end



sli.do: 3497821

How do we iterate through the Linked List?

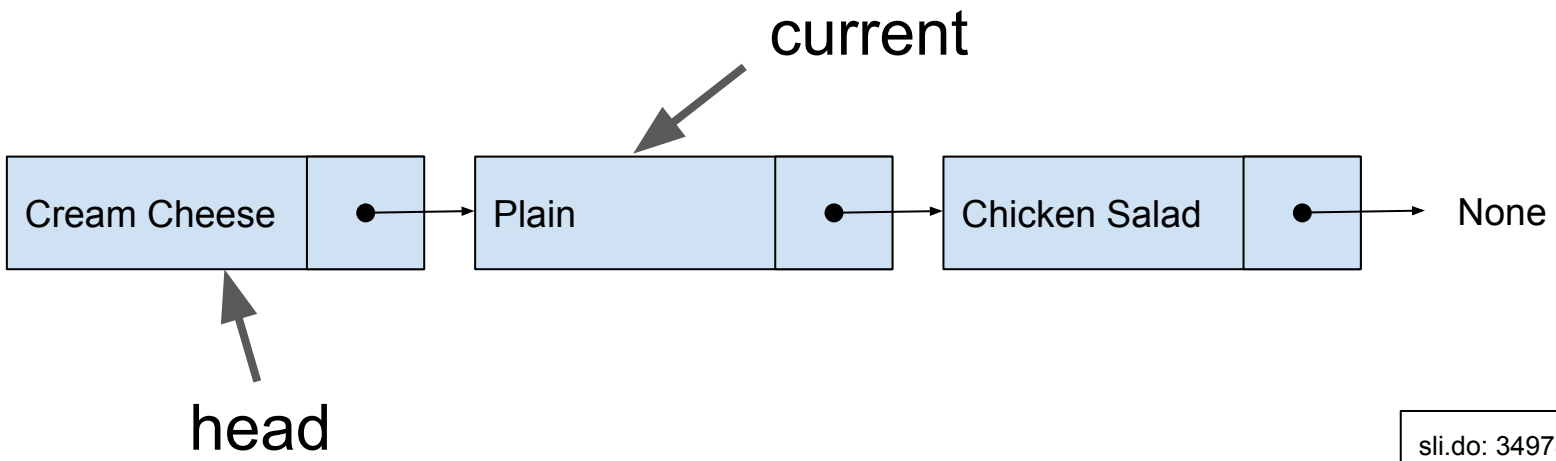
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end



sli.do: 3497821

How do we iterate through the Linked List?

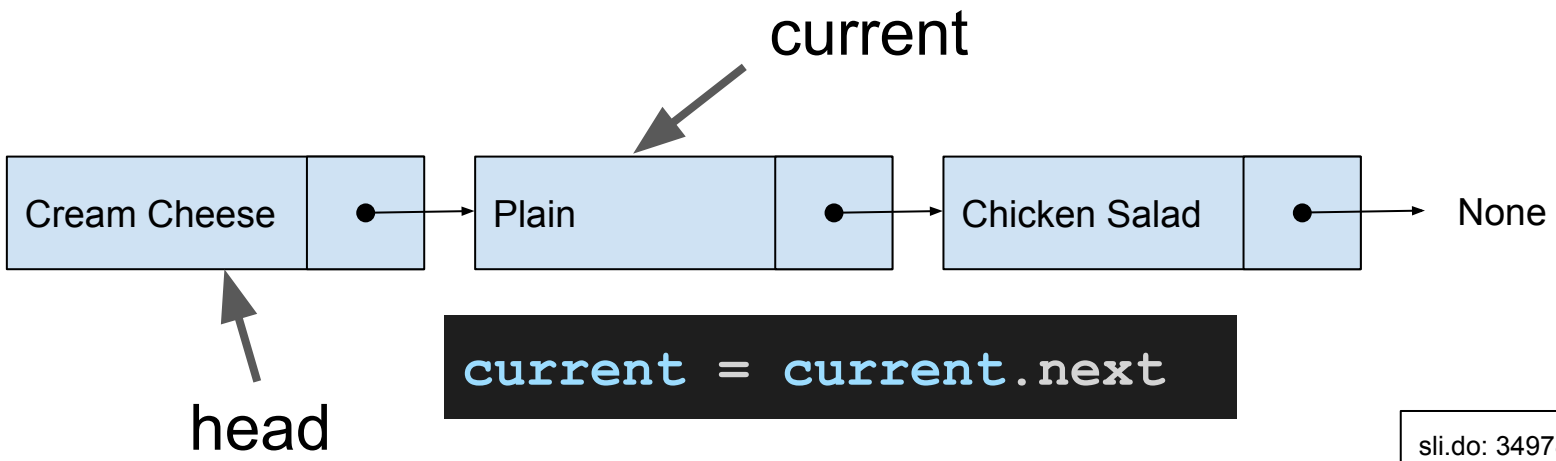
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end



sli.do: 3497821

How do we iterate through the Linked List?

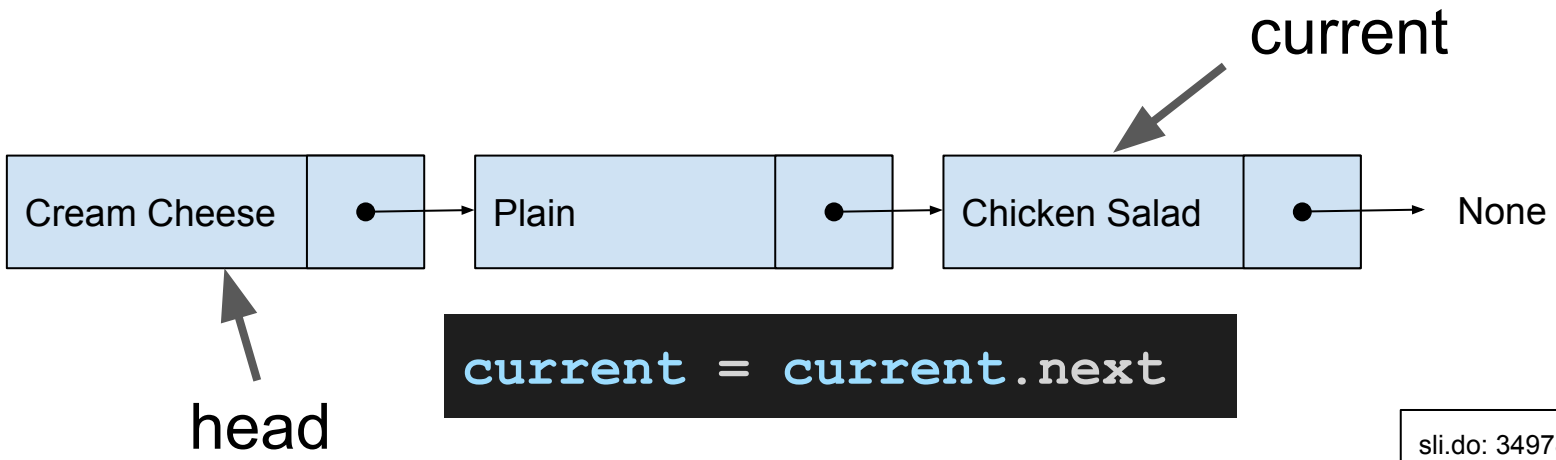
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end



sli.do: 3497821

How do we iterate through the Linked List?

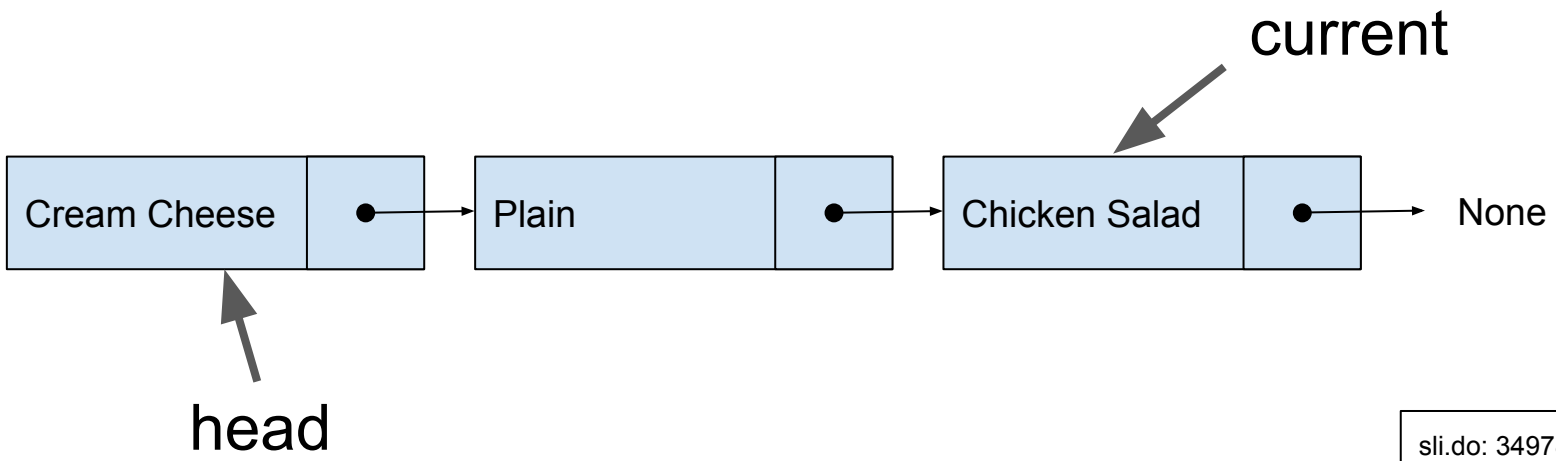
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end



sli.do: 3497821

How do we iterate through the Linked List?

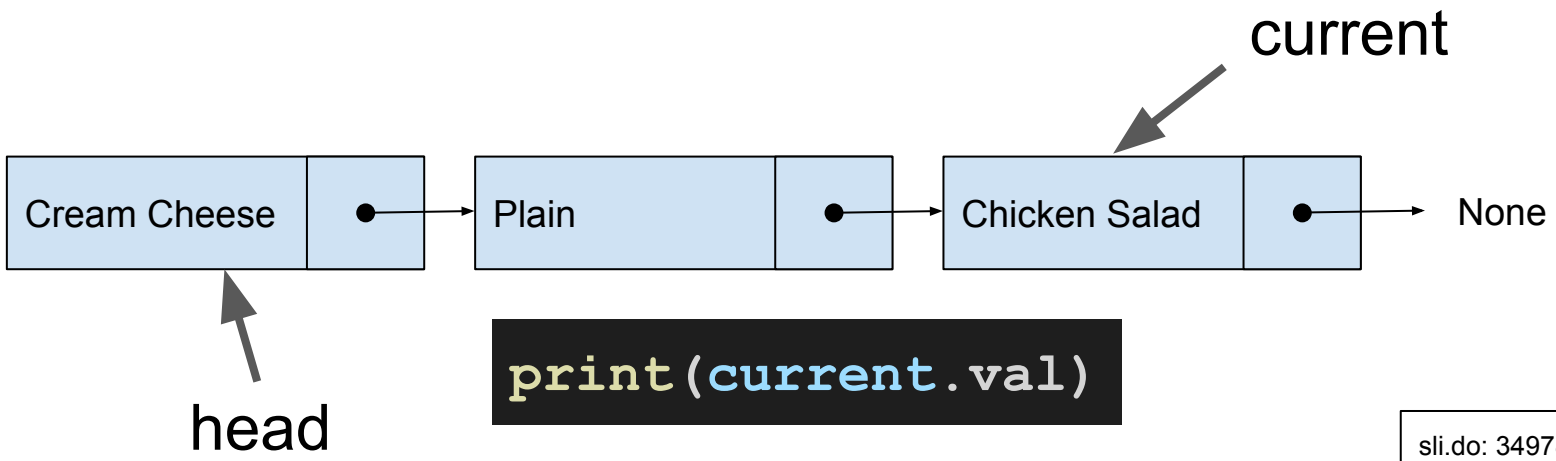
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end



sli.do: 3497821

How do we iterate through the Linked List?

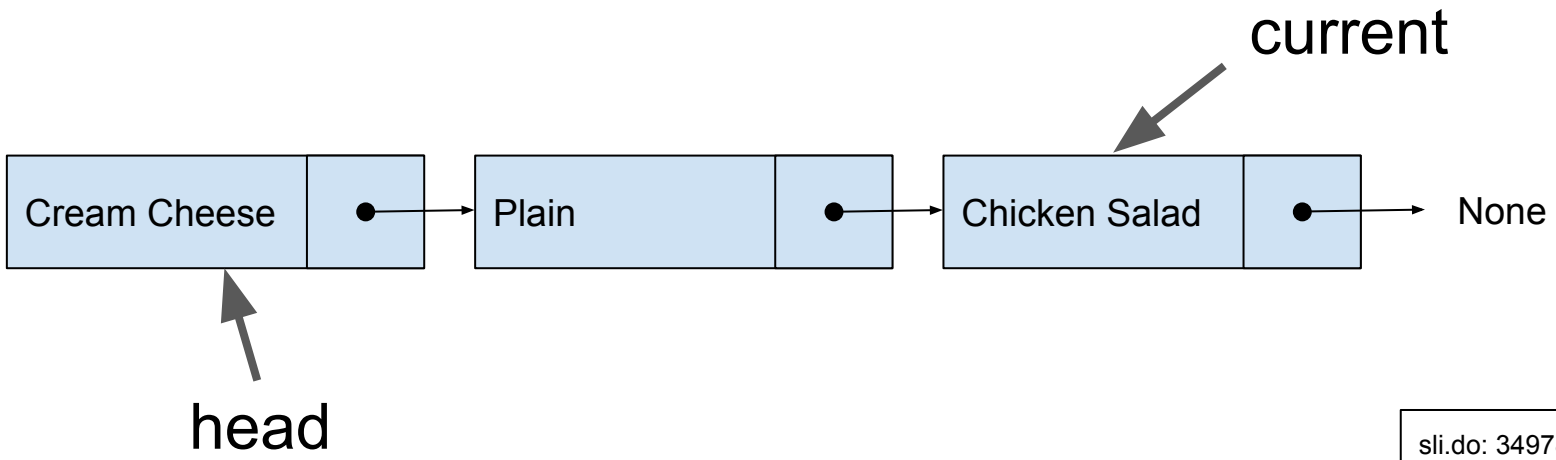
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end



sli.do: 3497821

How do we iterate through the Linked List?

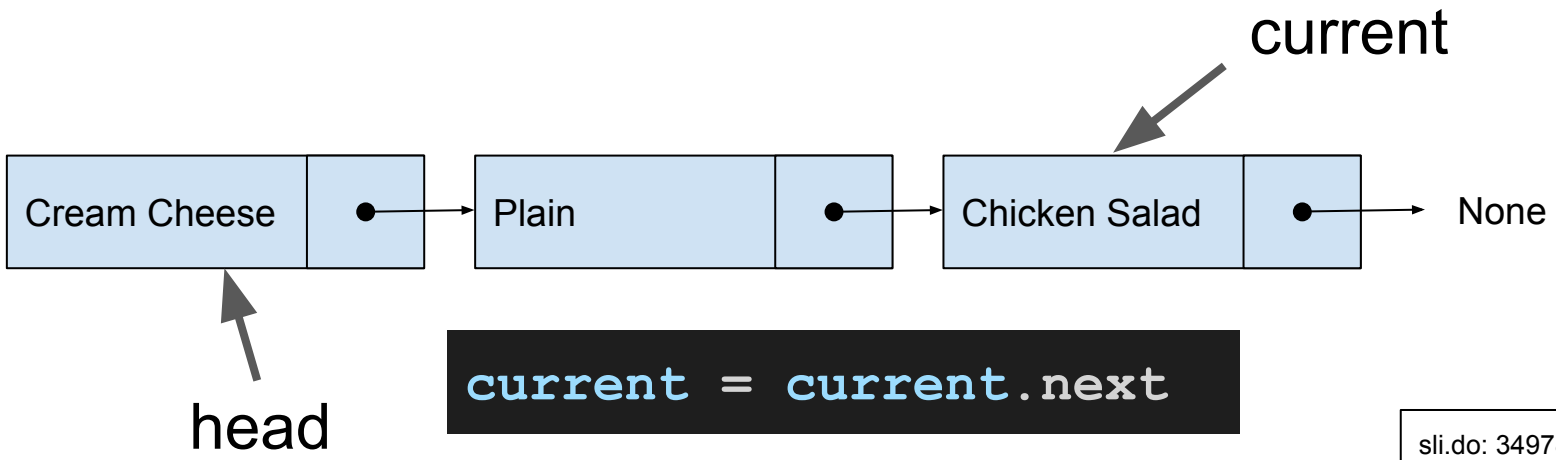
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end



sli.do: 3497821

How do we iterate through the Linked List?

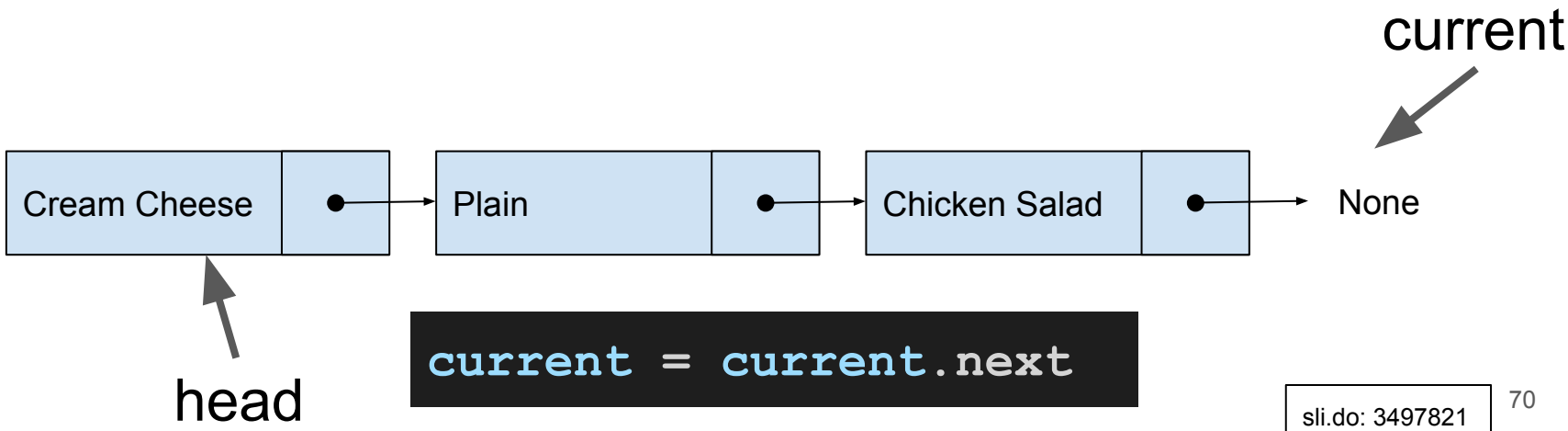
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end



sli.do: 3497821

How do we iterate through the Linked List?

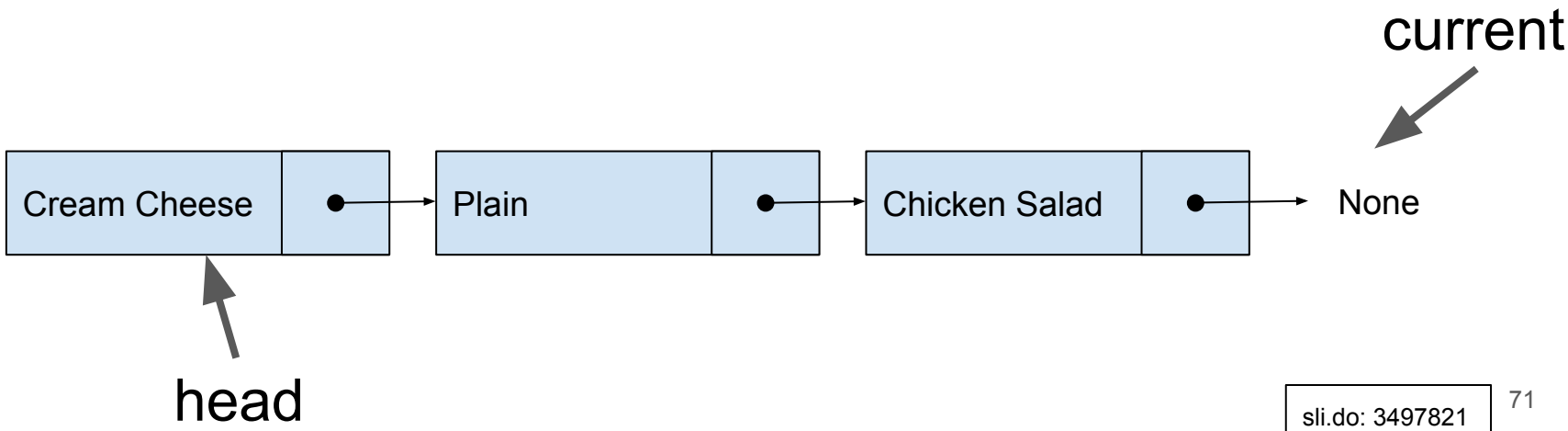
1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end



sli.do: 3497821

How do we iterate through the Linked List?

1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end

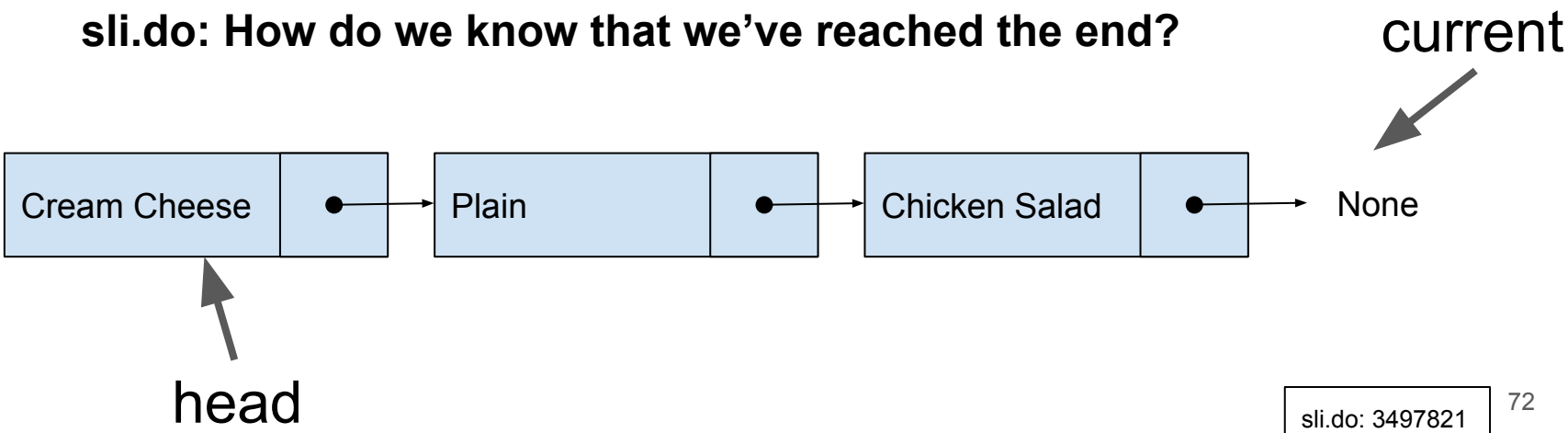


sli.do: 3497821

How do we iterate through the Linked List?

1. Make a new reference (“current”) at the start of the list
2. Print out “current”
3. Go to the next node
4. Repeat 2 and 3 until the end

sli.do: How do we know that we’ve reached the end?



sli.do: 3497821

Live Coding

sli.do: 3497821

Runtime - Sli.do

- Adding a Node to the front:
- Removing a Node from the front:
- Iterating through the entire linked list:

sli.do: 3497821

Runtime - Sli.do

- Adding a Node to the front: *O(1)*
- Removing a Node from the front:
- Iterating through the entire linked list:

sli.do: 3497821

Runtime - Sli.do

- Adding a Node to the front: *O(1)*
- Removing a Node from the front: *O(1)*
- Iterating through the entire linked list:

sli.do: 3497821

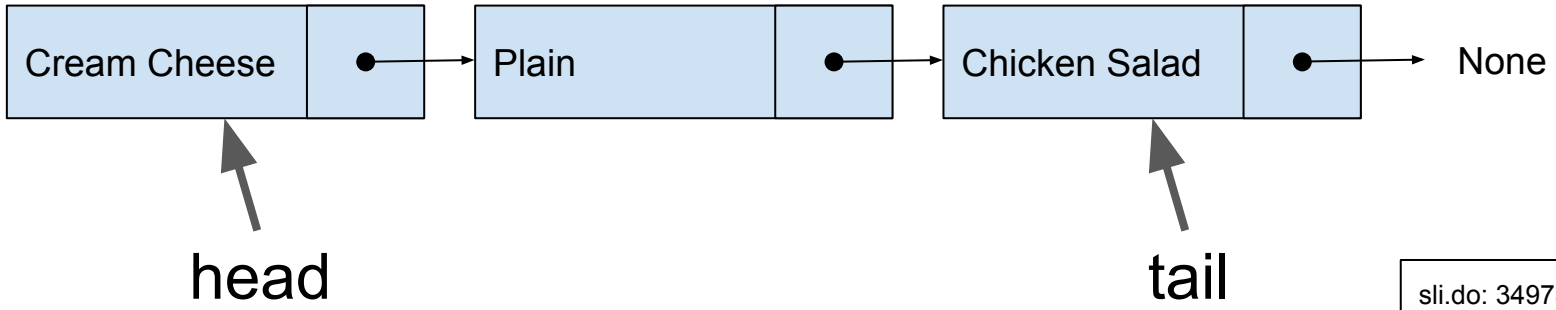
Runtime - Sli.do

- Adding a Node to the front: *O(1)*
- Removing a Node from the front: *O(1)*
- Iterating through the entire linked list: *O(n)*

sli.do: 3497821

Adding a tail

- Reference to the end of the list
- What are the benefits of having a reference to the end of the list?
- This will be your PA



sli.do: 3497821

Have a great weekend :)

sli.do: 3497821

Following lectures

- Talk about different Linked List variants and their runtimes (after they did the PA)
- Link back to Link Lists when covering other data structures (Dictionary chaining and stacks and queues)

sli.do: 3497821

PA / Lab

- Adding a tail to the Linked List
- Making the Linked List doubly linked
- Adding insert at position x and remove from position y
- Other aggregation functions over the list
- Keeping track of list length

sli.do: 3497821

Exam Questions

- Big O related Linked List questions
- Ask when to use regular lists vs Linked Lists
- Fill in the blank coding questions

sli.do: 3497821

Example Exam Question

Write code in the blank to complete the function

```
def add_front(self, val):
    '''
    Adds a new list node to the front of the linked list
    '''
    new_node = ListNode(val)
    
    self.head = new_node
```

sli.do: 3497821

Common Difficulties

- Forgetting to update “current” and “head” and “tail” reference
- Forgetting to call “.val”

Ways to Address:

- Exam review
- Review in following lectures
- Sli.do questions

sli.do: 3497821

Example Sli.do

What's missing?

```
def add_front(self, val):
    '''
    Adds a new list node to the front of the linked list
    '''
    new_node = ListNode(val)
    new_node.next = self.head
```

sli.do: 3497821

Example Sli.do

What's missing?

```
def add_front(self, val):
    '''
    Adds a new list node to the front of the linked list
    '''
    new_node = ListNode(val)
    new_node.next = self.head
    self.head = new_node
```

sli.do: 3497821

Teaching Philosophy

- Making students feel comfortable in class
- Adapting to students with diverse backgrounds

sli.do: 3497821

Teaching Philosophy

- Making students feel comfortable in class
- Adapting to students with diverse backgrounds
  - Sli.do for anonymous questions/polling

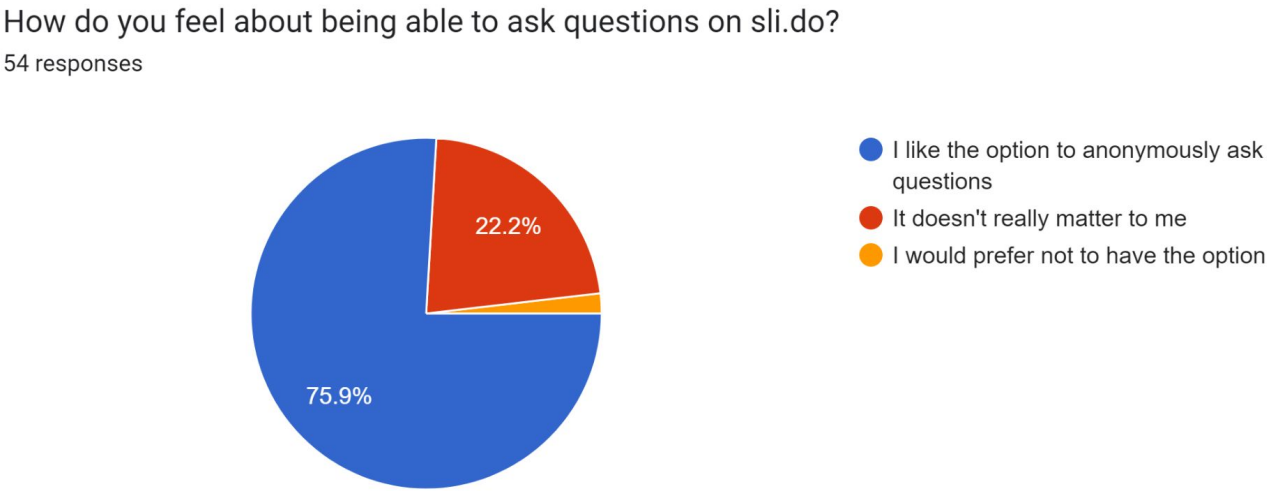
“He had a slido which allowed students to type questions anonymously which I felt allowed everyone to feel comfortable. I know a lot people like myself don't want to raise their hand in a big lecture class to ask a question so this was perfect.”

“He created a format where questions could be asked anonymously during class time, which allowed people to express confusion without fear of judgement.”

sli.do: 3497821

Teaching Philosophy

- Making students feel comfortable in class
- Adapting to students with diverse backgrounds
  - Sli.do for anonymous questions/polling



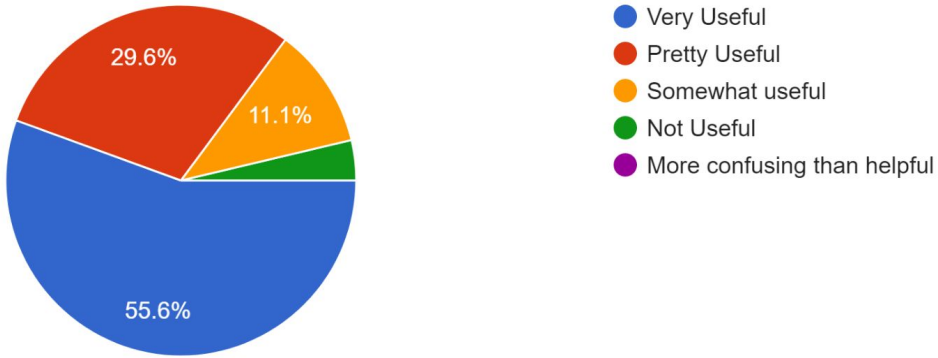
sli.do: 3497821

Teaching Philosophy

- Making students feel comfortable in class
- Adapting to students with diverse backgrounds
  - Sli.do for anonymous questions/polling

How helpful do you find the the instructor's asking "test" questions on sli.do?

54 responses



sli.do: 3497821

Teaching Philosophy

- Making students feel comfortable in class
- Adapting to students with diverse backgrounds
  - Sli.do for anonymous questions/polling
  - Offer Zoom lectures
  - Record lectures

“The recorded videos were essential because the material is so tough that I need to hear it multiple times.”

“I really like how Aidan accommodated for anything and everyone. I loved how we had the option to attend lecture real–time in person, real time over zoom, or recorded lectures.”

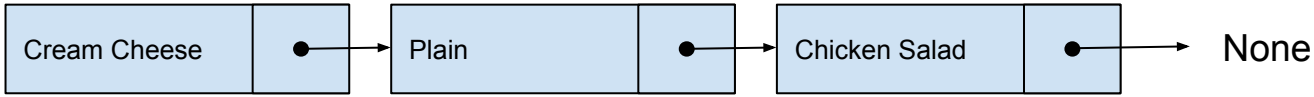
sli.do: 3497821

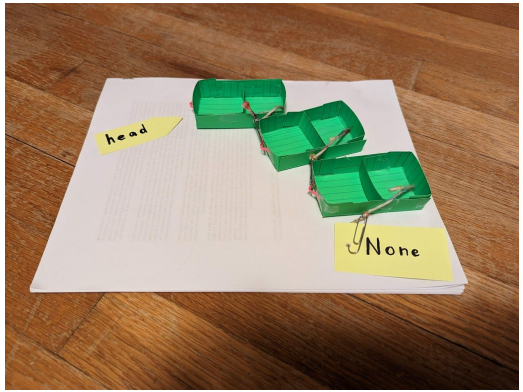
Teaching Philosophy

- Making students feel comfortable in class
- Adapting to students with diverse backgrounds
  - Sli.do for anonymous questions/polling
  - Offer Zoom lectures
  - Record lectures
  - Optional attendance
  - Have slides available ahead of time

sli.do: 3497821

Different learning styles






```
class ListNode:
    def __init__(self, val):
        self.val = val
        self.next = None
```

sli.do: 3497821

References

Linked List References: https://markfloryan.github.io/dsa1/slides/02-lists.html#/linkedlists, https://courses.engr.illinois.edu/cs225/sp2023/assets/lectures/slides/cs225sp23-05-linked-slides.pdf, https://www.cs.swarthmore.edu/~knerr/teaching/f16/topics/linkedlists.html

Chef Hat Icon: https://www.flaticon.com/free-icon/chef_481486?term=chef+hat&page=1&position=1&origin=search&related_id=481486

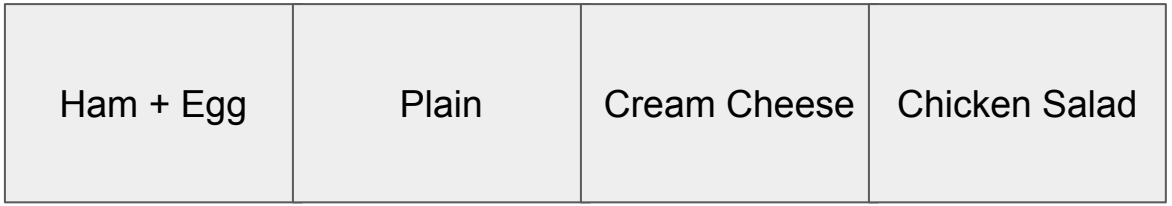
Bagel Images: https://www.newyorker.com/humor/daily-shouts/how-to-get-the-exact-amount-of-cream-cheese-you-want-on-your-bagel, https://www.nybagelsandbuns.com/Plain_Bagel_p/plain.htm, https://www.fareway.com/recipes/everything-but-the-bagel-chicken-salad, https://oakmontbakery.com/product/breakfast-sandwich-ham/

Taylor Swift Image: https://www.billboard.com/music/music-news/taylor-swift-signed-guitar-auction-1235181366/

Bodos's Bagels Image: http://www.bodosbagels.com/home.html

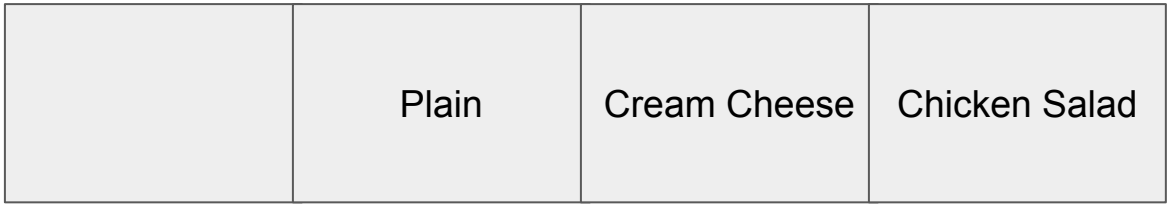
sli.do: 3497821

Regular Python List Removal from Front



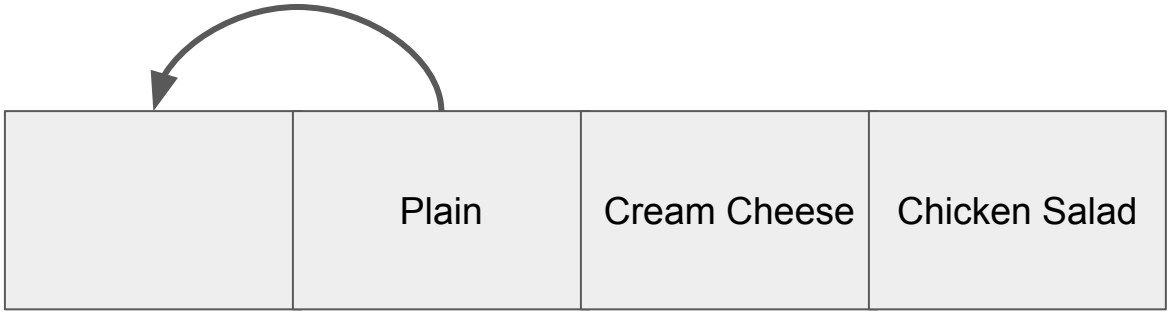
sli.do: 3497821

Regular Python List Removal from Front



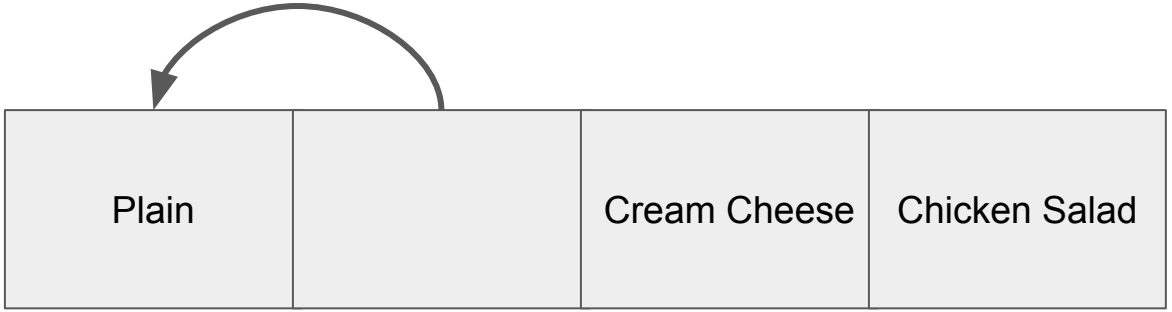
sli.do: 3497821

Regular Python List Removal from Front



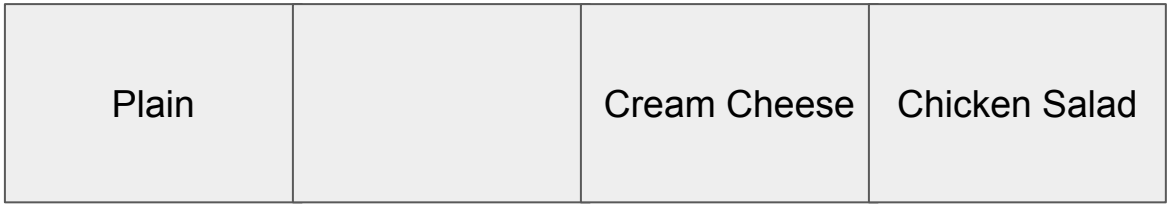
sli.do: 3497821

Regular Python List Removal from Front



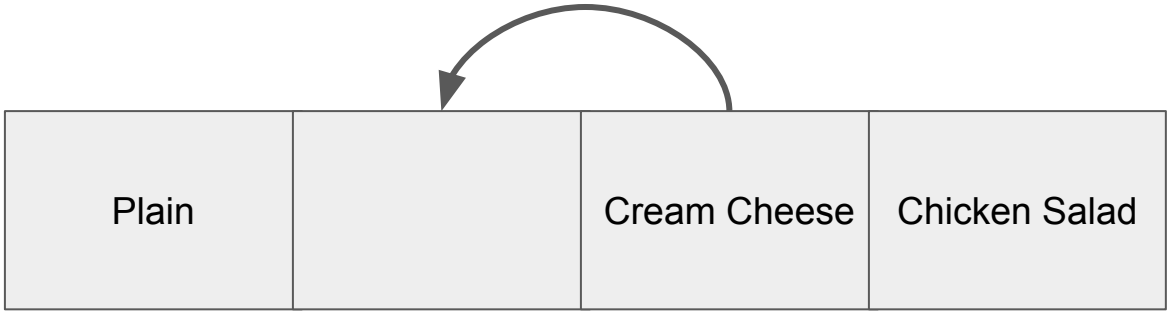
sli.do: 3497821

Regular Python List Removal from Front



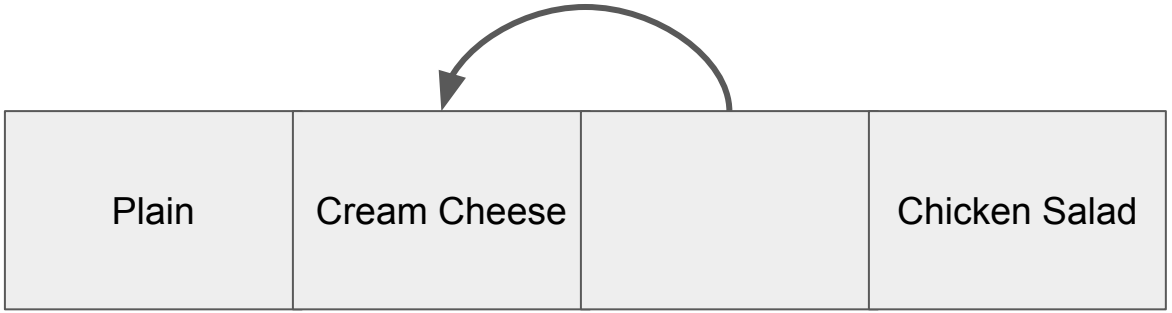
sli.do: 3497821

Regular Python List Removal from Front



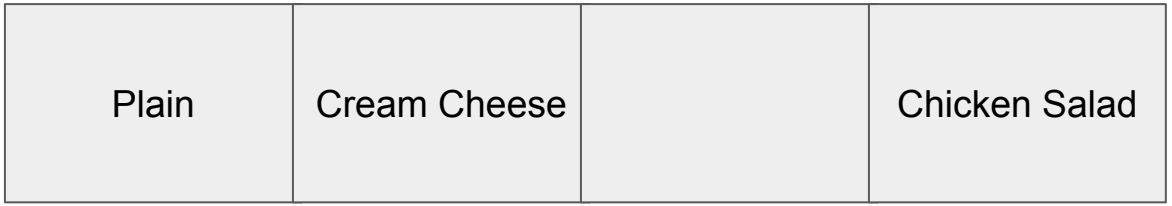
sli.do: 3497821

Regular Python List Removal from Front



sli.do: 3497821

Regular Python List Removal from Front



sli.do: 3497821

Regular Python List Removal from Front

sli.do: 3497821

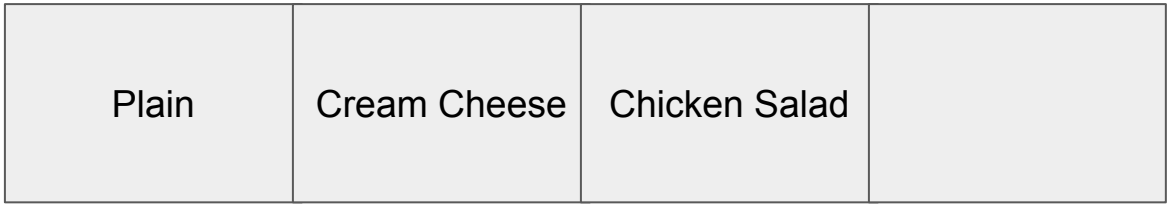
Regular Python List Removal from Front

sli.do: 3497821

Regular Python List Removal from Front

sli.do: 3497821

Regular Python List Removal from Front



sli.do: 3497821