Algorithms and Data Structures for Data Science

Lists

CS 277
Brad Solomon

September 13, 2021

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN
Department of Computer Science

lab_cipher Feedback

Average completion time looks to be ~2 hours

Pretty happy with outcome of lab overall

Some people don’t like the 20 minute lecture.

Avg: 18 / 15

lab_cipher review

Plaintext:	D	D	D	B	B	D
Ciphertext:						

A	0
B	1
C	2
D	3
E	4

Course Pacing Feedback

Most of class actually overqualified for class

Everyone thinks pacing is fine

Everyone thinks content is either fine or too basic

mp_racing was too hard

Learning Objectives

Discuss implementation strategies for lists

Debate which strategy is better (and how to judge ‘better’)

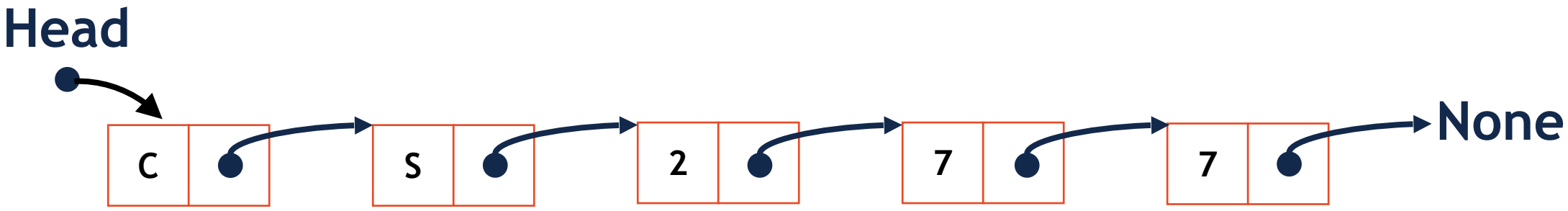
Introduce the concept of asymptotic efficiency

List Implementations

1. **Linked list**

2.

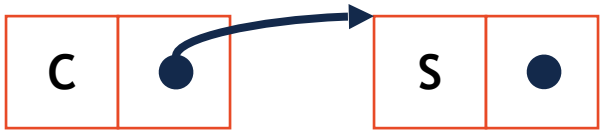
Linked List




```
class Node:
    def __init__(self, data, next=None):
        self.data = data
        self.next = next

class linkedList:
    def __init__(self, head=None):
        self.head = head
```

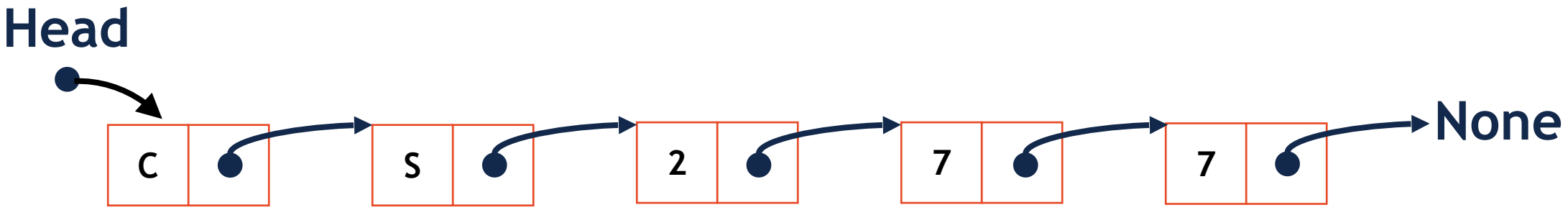
Linked List Add

```
class Node:
    def __init__(self, data, next=None):
        self.data = data
        self.next = next

class linkedList:
    def __init__(self, head=None):
        self.head = head

    def add(self, data):
        temp = self.head
        self.head = Node(data, temp)




if __name__ == '__main__':
    ll = linkedList()

    for i in range(3):
        ll.add(i)


```



Linked List

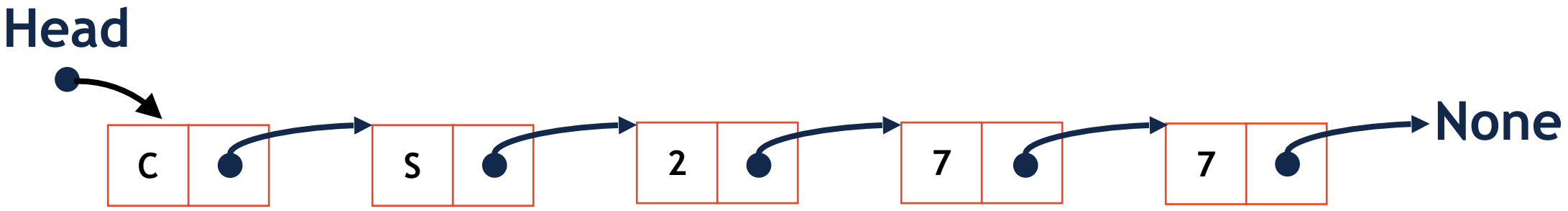
Index(3)




```
class linkedList:

    def index(self, pos):
        curr = self.head

        i = 0
        while(curr and i < pos):
            curr = curr.next
            i+=1

        if i == pos:
            return curr

```

Linked List

Insert(X,2)

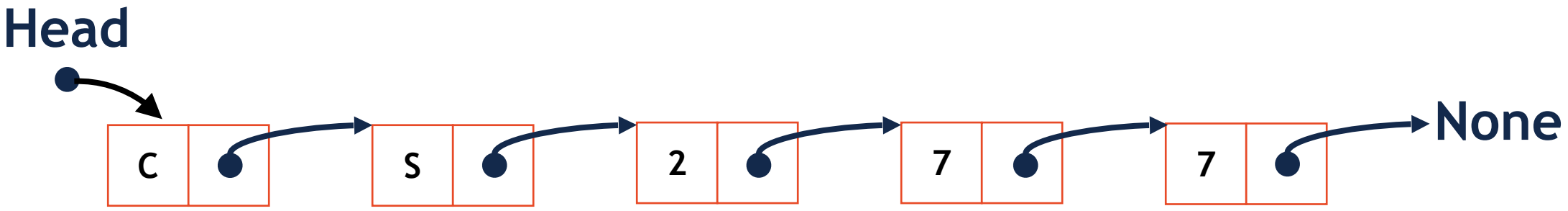


Linked List Insert

```
class linkedList:

    def insert(self, data, pos=0):
        if (pos == 0):
            self.add(data)
        else:
```

Linked List

Remove(7)

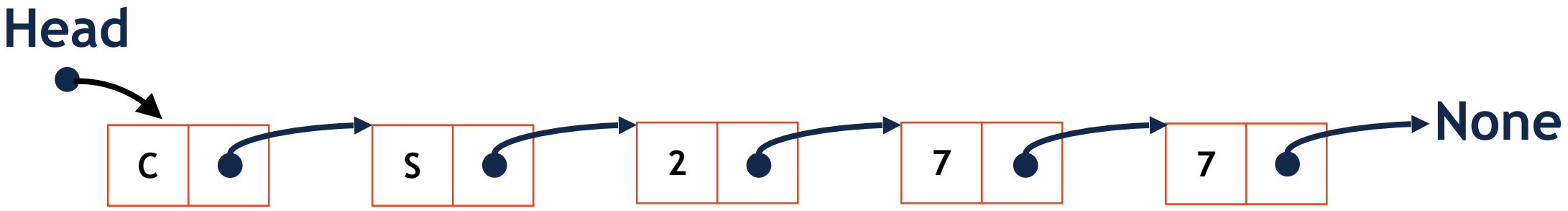


Linked List Remove

```
class linkedList:
    def remove(self, data):
        prev = None
        curr = self.head
```

Linked List

Pop(4)



Linked List Pop

```
class linkedList:
   def pop(self, pos):
```

Exercise 1

```
ll = linkedList()

for i in range(3):
    ll.add(i)


ll.insert(6,1)
ll.insert(3,1)
ll.insert(10,3)


ll.remove(6)
ll.remove(2)
ll.remove(12)


ll.pop(2)
ll.pop(0)
```

List Functions

Index:

Add:

Insert:

Remove:

Pop:

List Implementations

1. Linked List

2. Arrays

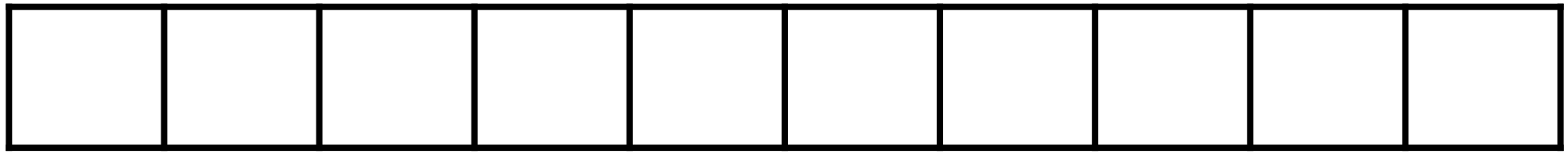
Array

Creating an array list

```
1 class arrayList:
2     def __init__(self, array=[]):
3         self.array = array
```

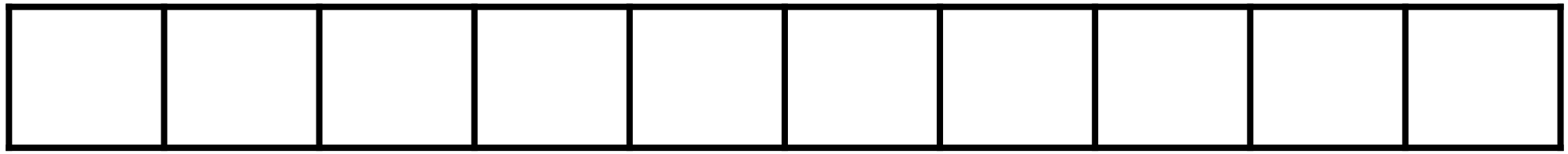
Array Add

```
1 def add(self, data):
2
3
```



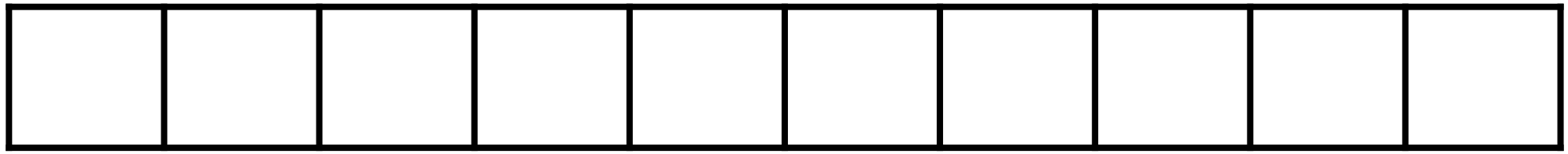
Array Index

```
1 def index(self, pos):
2
3
```



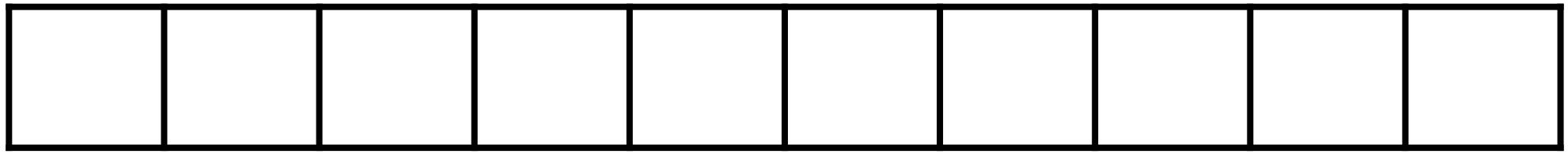
Array Insert

```
def insert(self, data, pos=0):


```



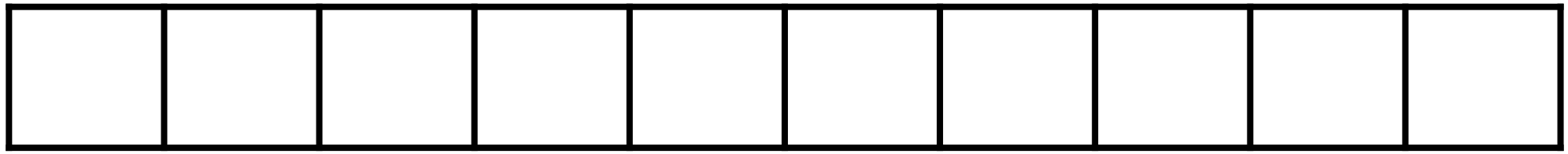
Array Remove

```
def remove(self, data):


```



Array Pop

```
def pop(self, pos):


```



Which implementation is better?

What do we care about when we write code?