Data Examples

Announcements

Midterm Regrades out today.

Rally on Sproul on 4/8. A *lot* of rallies/protests over the next few weeks.

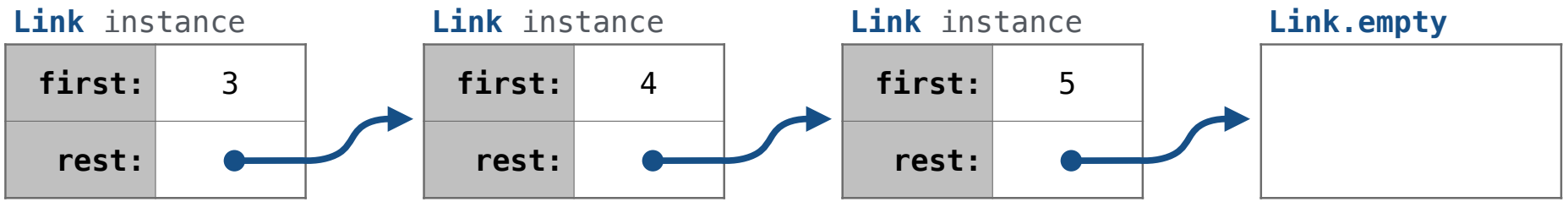
Ants project will be coming soon

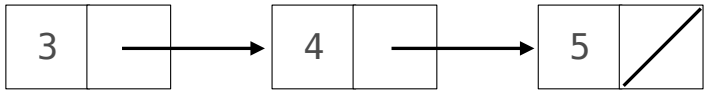
Find partners; start early

Linked Lists Practice

Linked List Notation

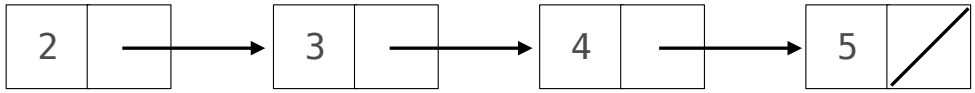
```
s = Link(3, Link(4, Link(5)))
```



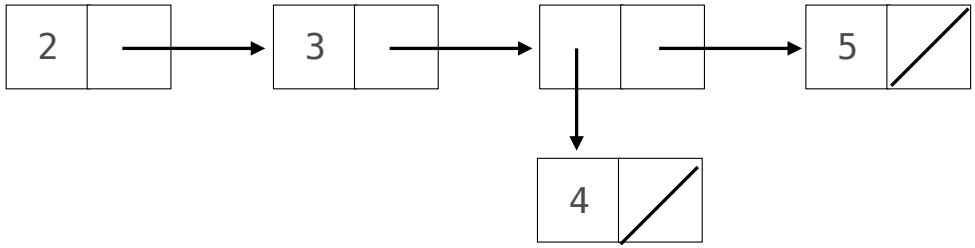


Nested Linked Lists

```
>>> s = Link(2, Link(3, Link(    4     , Link(5))))
>>> t = Link(2, Link(3, Link(  Link(4) , Link(5))))
>>> print(s)
<2 3 4 5>
```

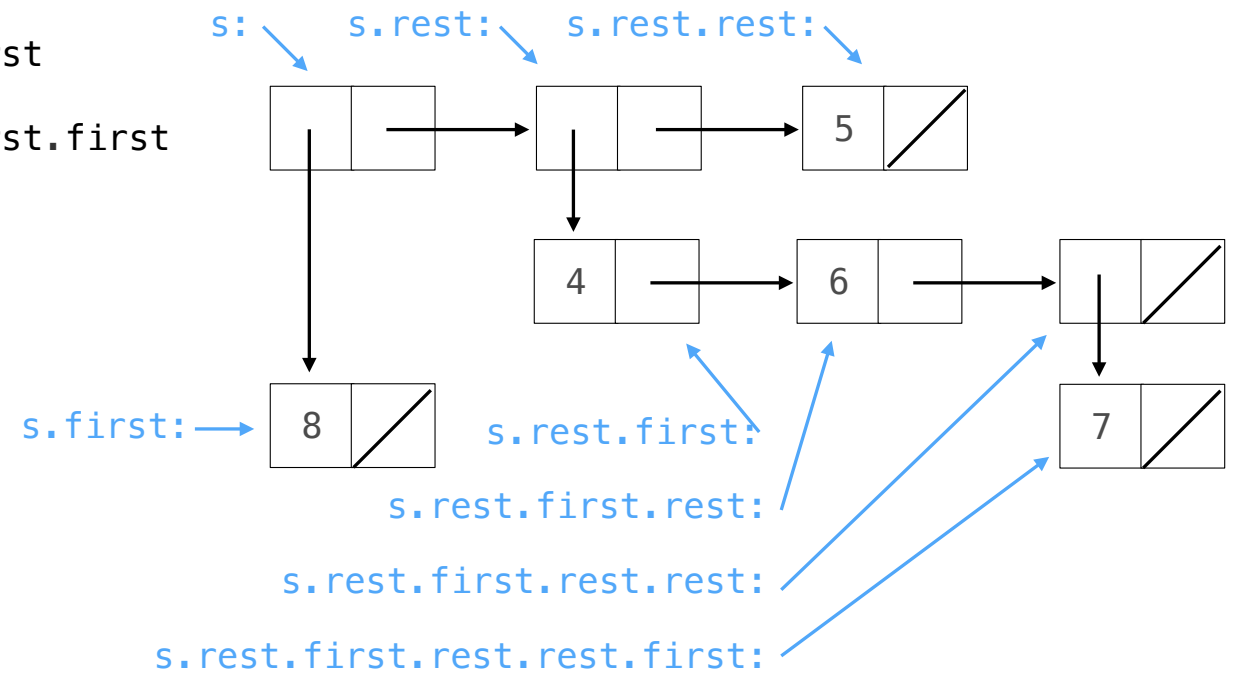



```
>>> print(t)
<2 3 <4> 5>
```



Nested Linked Lists

```
>>> s = Link(Link(8), Link(Link(4, Link(6, Link(Link(7)))), Link(5)))
>>> print(s)
<<8> <4 6 <7>> 5>
>>> s.first.first
8
>>> s.rest.first.rest.rest.first
Link(7)
>>> s.rest.first.rest.rest.first.first
7
```



Linked Lists Can Contain Anything:

What if we make an Album, which is a linked list of songs?

```
fearless = Song("Fearless", "Taylor Swift", 241)

fifteen = Song("Fifteen", "Taylor Swift", 294)

fearless_album = Link(fearless, Link(fifteen) … )
```

How can we…

Find the longest song?

Finding the Longest Song

Given a linked list of Songs, which Song is the longest?

```
def total_album_length(album):
    if album == Link.empty:
        return None
    if album.rest == Link.empty:
        return album.first.length

    rest_longest = longest_song(album.rest)
    if album.first.length > rest_longest.length:
        return album.first
    else:
        return rest_longest
```

Object-Oriented Programming Practice

OOP Design Tips

Inheritance works well when two things have a *hierarchical* relationship.

Otherwise, we can *compose* classes together.

Two useful facts:

* Using __ (‘double underscore’) as a prefix for a variable name or method makes it “private”

* Use _ (single underscore) as a prefix for a variable name is “just” communication that a variable is “internal”

Revisiting Bank Accounts

We had Checking Accounts, Savings Accounts

* Some have savings rates that accrue interest

* We need to maintain a list of accounts

* Now each account needs an account number

How should our old model evolve?

Should SavingsAccount inherit from CheckingAccount?

What if we make a new class called Bank?

Should CheckingAccount inherit from the Bank class?

Exploring Our Bank…

Try to make a new Bank

Poke at which methods are public vs private

Future Notes: What's Missing From C88C?

We ***barely*** scratch the surface of Python features:

* classes can have 'static' attributes and methods
* many different magic methods exist: https://docs.python.org/3/reference/datamodel.html
* classes can inherit from more than 1 class at once

There's lots to learn about OOP. More will come in CS 61B.