Introductory Computer Science
Week 7 – The Big Reveal & Beyond

Dann Sioson

dj.sioson@alum.utoronto.ca

September 8, 2019

Administrative items

- You're probably wondering why this week is a PowerPoint (again) instead of a pdf
- Exercises*
- Assignment

* = Depends on the current situation

Itinerary

1. ~~Introduction and Git~~
2. ~~Programming With Python*~~
3. ~~Memory Model and Debugging*~~
4. ~~Object Oriented Programming~~
5. ~~Object Oriented Programming~~
6. ~~Test day~~
7. The Big Reveal & Beyond

The big reveal

- You ready for the big reveal?
- This is gonna blow your mind
- No like actually

LISTS EXIST

DON’T

The Big Reveal

- Lists aren't possible
- They're just an ADT (Abstract Data Type)
- "… The greatest trick Python ever pulled was convincing the world lists exist"

Why is this the case?

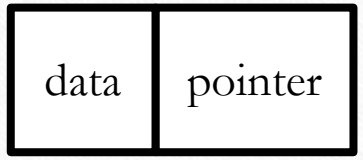
- As a Computer Scientist, you want to have things as efficient as possible
- The memory model is a bit of a lie. That is, each id in the memory stack can only call out how much memory it needs beforehand
  - Tuples can easily do this
  - Lists can't (i.e. inserting, appending, and removing)
  - … didn't we say in week 2 that tuples are not mutable, and that lists are mutable!?

We're going to create a list

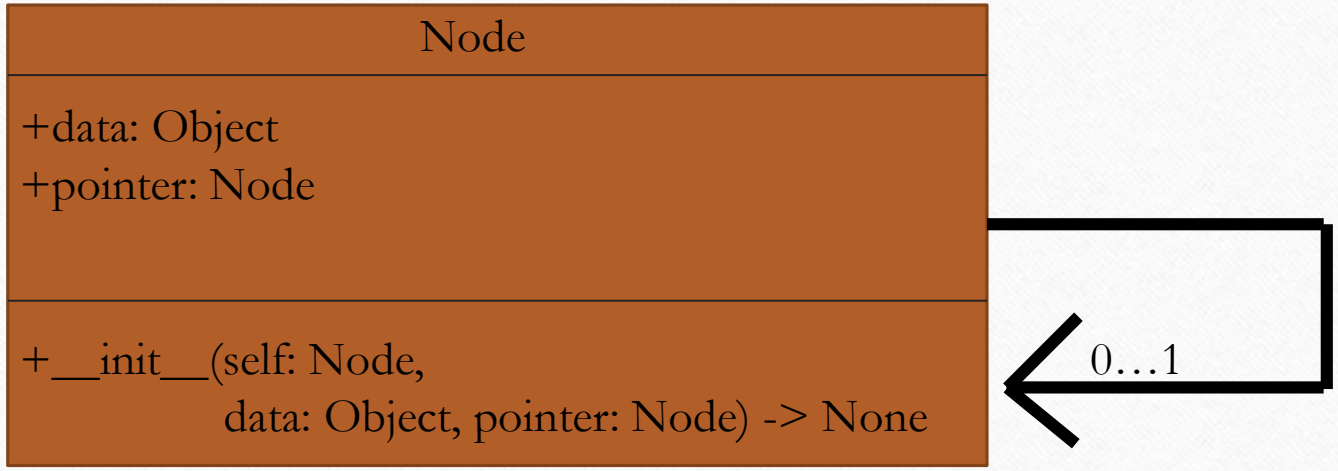
- … any ideas?
- What would be the problem if we say "allocate 100 blocks of memory for this 'list'?"
- Hint: Maybe a data structure (via OOP)?

Base structure: Node

- A node holds data
- A node has a pointer, which references something

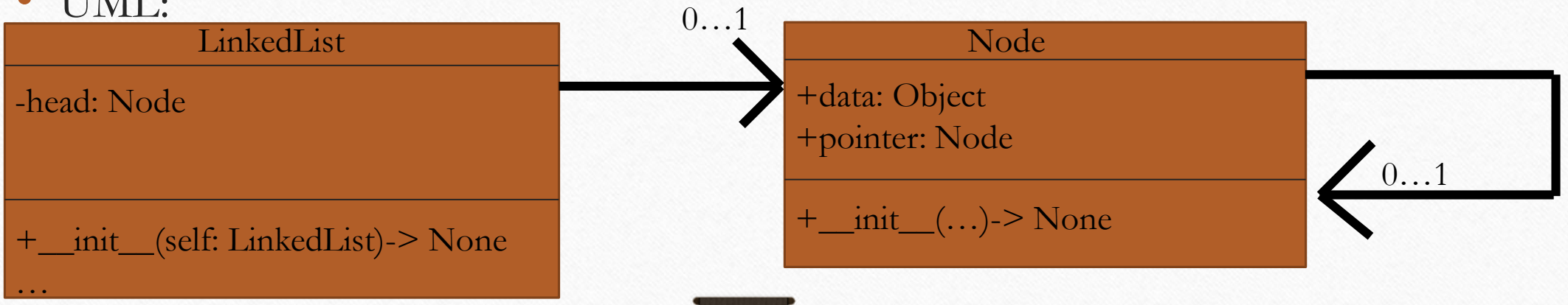


UML of a Node (at base)



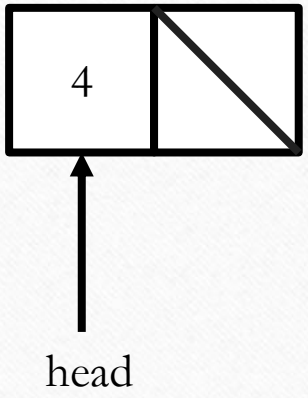
Linked List

- Typically only has one main feature, which is a head that may contain a Node
- However, linked lists can be further extended for other low level features (e.g. length of list, tail, middle, etc.)
- UML:

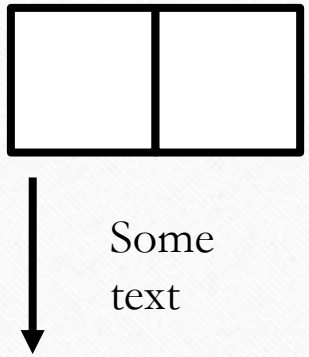


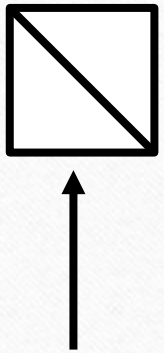
…What features can we add to a linked list?

Before we start, a bit of explanation with an example:

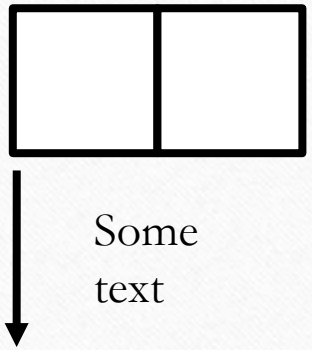


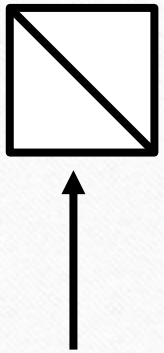
Prepending



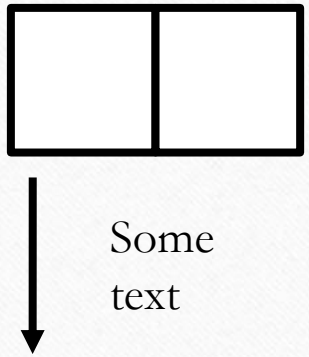


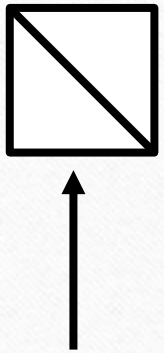
Appending

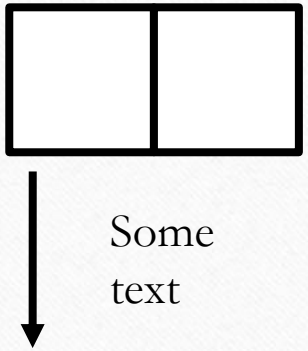




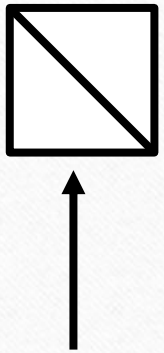
Inserting (assuming we don't run into index errors)







Removing (assuming we don't run into index errors)



Beyond!

With the materials that I have taught…

- There are things I did skip relatively due to time (and personal) constraints
- There are things that are important for you to know and can learn more about at your own time (highlighted in bold):
  - Following the design recipe, we did not do any tests in OOP. This is because you would have to **unit test** your code in order to understand its behaviour
  - Like the last two examples that we did, we went into the assumption of no indexing errors. It's critical to know about **exception handling,** especially in structured languages
  - I wanted to have two weeks on **Recursion**, which is a neat tool to do certain things if you don't like the iterative approach using loops; useful in University (especially for proofs)
  - Linked Lists is not the only **data structure** that is important
  - As a SWE, you're going to be doing a lot of design; look at **design patterns** and **SOLID principles** to reduce code as much as possible

Thank you for the opportunity to let me teach!