ISTA 350

•••

Linked Lists and While Loops
Spring 2020

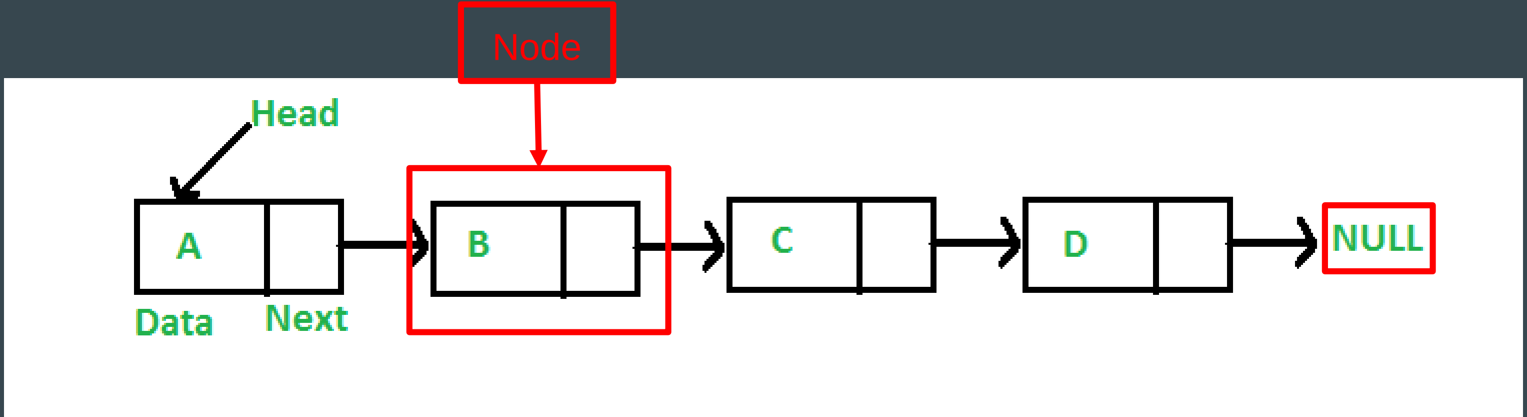
Last time...

- Classes review
- Regular Expressions

Linked Lists

Linked Lists

A linked list is a sequence of data elements, connected together via links. Each data element contains a connection to another data element in the form of a pointer.



`Head` is the first element in your linked list.

Each node contains `data` and a link to the `next` node.

The linked list ends when `next == None`.

Creating a Linked List

We create a linked list by creating a `Node` class and then another class (our `LinkedList` class) that uses the `Node` object.

```
class Node:
  def __init__(self, datum = None):
    self.datum = datum
    self.next = None
class LinkedList:
  def __init__(self):
    self.head = None
```

Example: Adding elements to a Linked List

```
>>> linkedlst = LinkedList()
>>> linkedlst.head = Node(“Hey”)
>>> el2 = Node(“350”)
>>> el3 = Node(“students!”)
>>> linkedlst.head.next = el2
>>> el2.next = el3

>>> linkedlst.to_list()
[“Hey”, “350”, “students!”]
```

Try drawing the structure of this linked list!

`while` Loops

`while` review: the basics

`while` loops execute a set of commands as long their condition is true.

For example,

```
i = 0
while i < 6:
  print(i)
  i += 1
```

In this example, we're using a while loop similar to how we would expect a simple `for` loop to work - by initializing `i` to zero and incrementing until it exceeds our condition (when `i` equals 6)

while review: break

With a break statement, we can stop the loop even if the while condition is still true.

```
i = 0
while i < 6:
  print(i)
  if i == 3:
    break
  i += 1
```

In this case, when i == 3, the loop will completely stop. What would the output look like?

If you return, that will also stop the loop.

while review: while True

while True is a type of while loop where the condition is never False. In order to stop the loop, we have to explicitly tell it to break or return. Otherwise, we’ll enter an infinite loop that never ends.

```
while True:
  user = input(“Enter a number 1-6 or ‘q’ to quit: “)
  if user == ‘q’:
    break
  print(user)
```

Traversing Linked Lists with `while` loops

Say we want to navigate to the last node in our linked list.

- `current` is the node we're on
- `current.next` is the next node, linked to `current`

```
current = <self/List>.head #start at the beginning
while current.next: #while current.next exists (is not None)
  current = current.next #current gets assigned current.next
```

The loop will end when `current.next == None`, i.e., when there isn't a `next` node. When we assign `current = current.next`, we're sort of moving to the next node in the linked list.

Complete lab02.py

The function `to_list` has been given to you for your convenience. It prints the contents of your linked list as a regular Python list. Use it to test your code!

A couple reminders:

- Keep trying your best!
- If you have questions, you can email us at the listserv!
  - **`ista350-questions@list.arizona.edu`**