AMACSS Study Group

Exceptions and Linked Lists

Exceptions

Why use exceptions?

- Allows us to create custom errors relevant to our program
- Can create more informative error messages that help us
- Part of a smart design, outside users can get helpful info about our code through the errors we raise
- Python built in errors might not help us solve a problem about our ADT
    - IndexError might not give us a clue on how to solve an issue with our Matrix class

How to define exceptions

- Exceptions are defined similarly to how regular class are define
- The difference is that they inherit the Exception class
    - Recall last weeks discussion on how Parent-Child relationships work

Sample exception

```
class MyError(Exception):

    def __init__(self):
        print(“This is my new exception”)
```

Exercise

Define and raise your own exception class that is raised when a user does something like the following:

```
    a = []
    print(a[3])
```

And another for when the user does this:

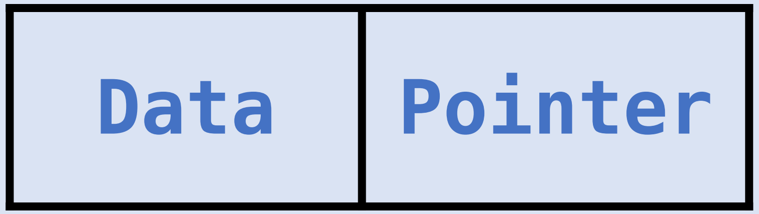
```
    1 = “Hello”
```

Linked Lists

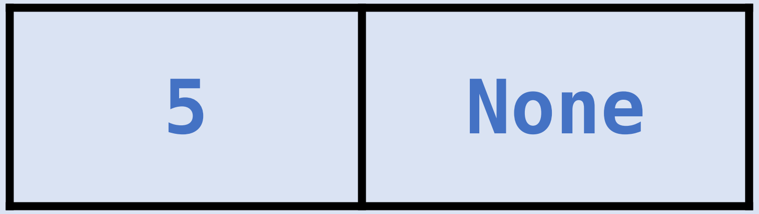
What are linked lists

- Linked lists are linked data structures that we use to store data
- Nodes are created, and each node is linked to the next one in the list
- Nodes have two “parts”: The pointer to the next node, and the node’s data
- The first node in a linked list is called the head of the list
- The data of a node could also be another pointer
  - A snakes and ladders board assignment from last term had a node with pointers to the top and bottoms of the snake/ladder, with no other data

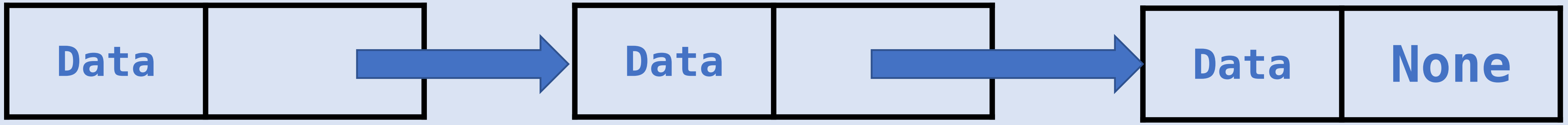
Graphic representation of a node in a linked list



Here is a node that stores the number 5 and doesn’t point to anything



Several nodes in a linked list

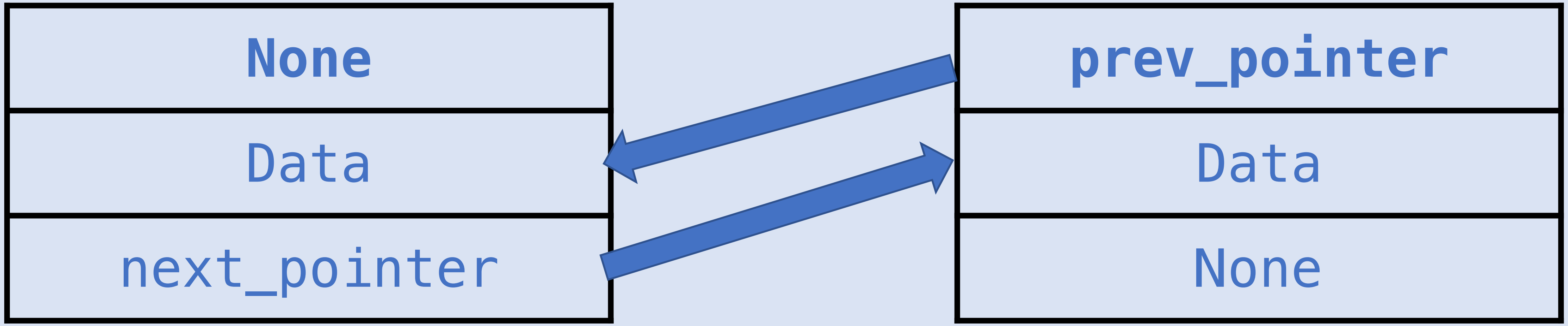


Data could be any value we assign

Lets look at an implementation of a linked list

Another kind of linked list

- A doubly linked list:
  - A doubly linked list is just like a regular linked list, except that it has an extra pointer that points to the previous node



Try the following:

Implement a doubly linked list with a minimum length of 5. Using a while loop, set the values of the nodes to numbers of your choosing

Hint: Use counters to track where in the list you are, and whether the next node is the end of the list or not