CSCI 132:
Basic Data Structures and Algorithms

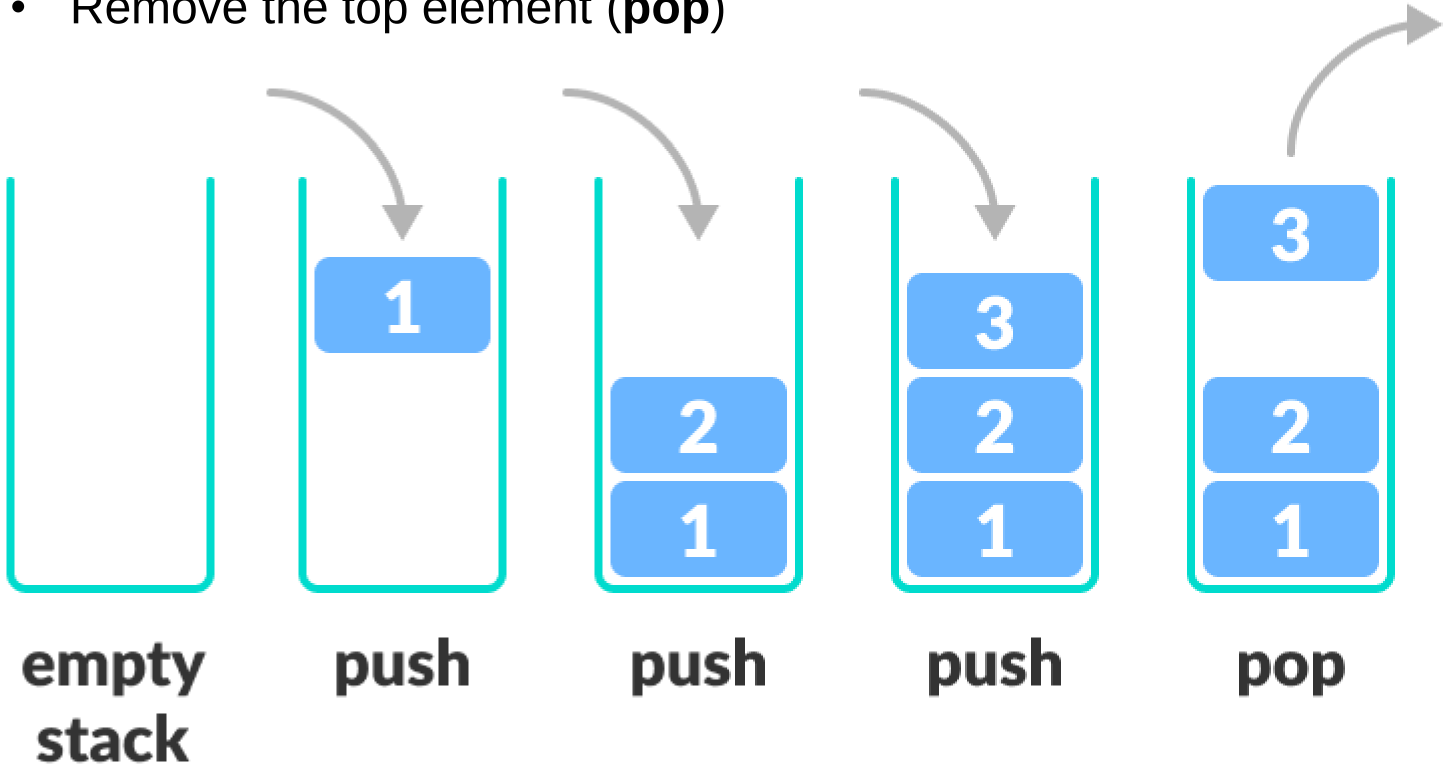
Stacks (Linked List implementation)

Reese Pearsall
Spring 2024

A **stack** is a data structure that can hold data, and follows the **last in first out (LIFO)** principle

We can:

- Add an element to the top of the stack (**push**)
- Remove the top element (**pop**)

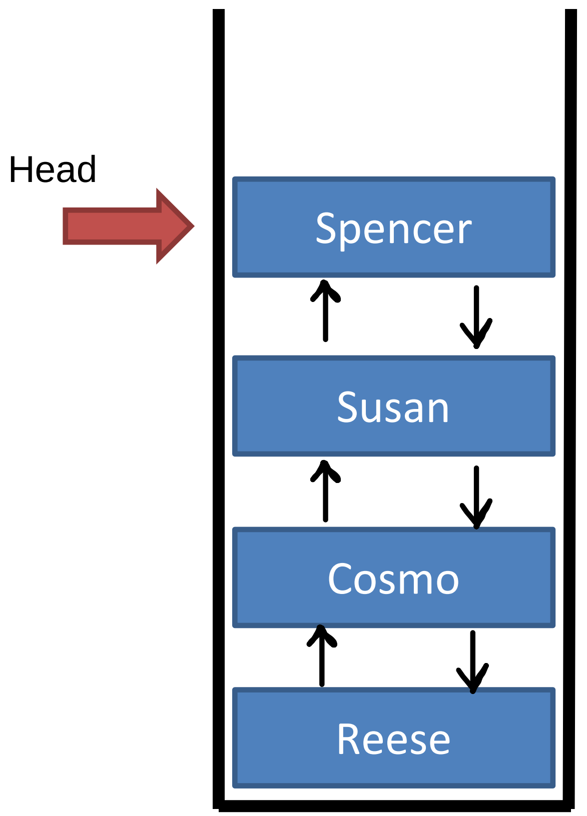


We can implement a Stack using an Array, or a linked List

Stack Implementation (Linked List)

To Do List:
- Push()
- Pop()
- Peek()
- IsEmpty()

We will import the Linked List Library (we will not write our own linked list class)



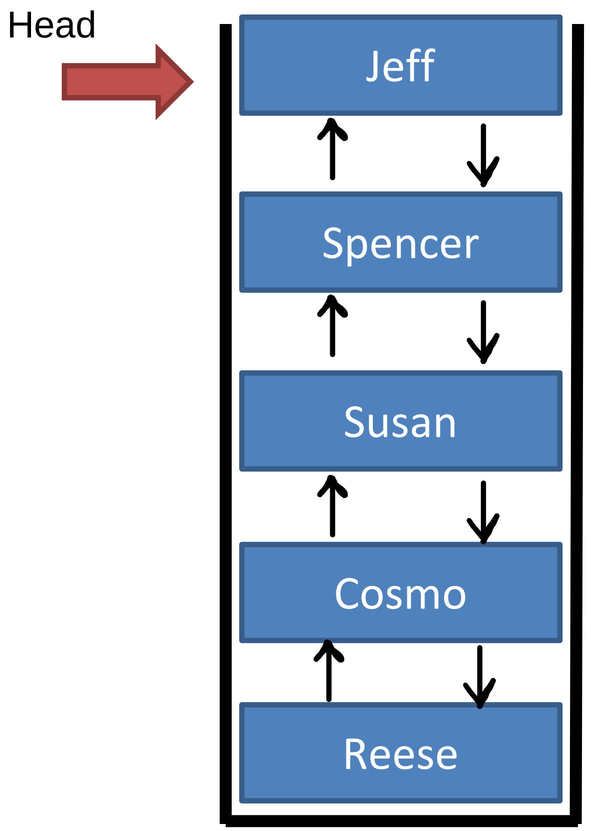
If we don’t know how big out stack needs to be ahead of time, then using a linked list will be a better choice than an array/arraylist

The top of the stack will be the head of the linked list

Stack Implementation (Linked List)

To Do List:
- Push()
- Pop()
- Peek()
- IsEmpty()

We will import the Linked List Library (we will not write our own linked list class)



```
stack.push("Jeff")
```

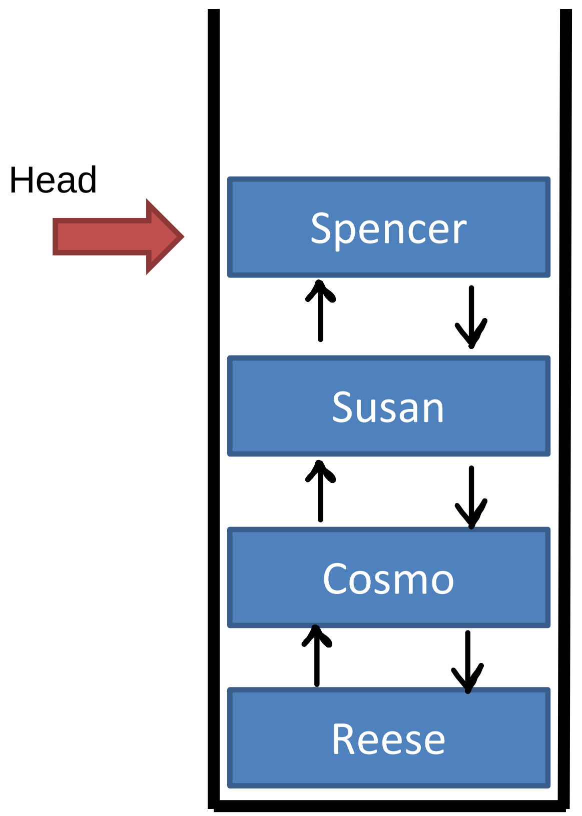
Whenever we add something to the stack, we add the element to the front of the linked list

The top of the stack will be the head of the linked list

Stack Implementation (Linked List)

To Do List:
- Push()
- Pop()
- Peek()
- IsEmpty()

We will import the Linked List Library (we will not write our own linked list class)



```
stack.pop()
```

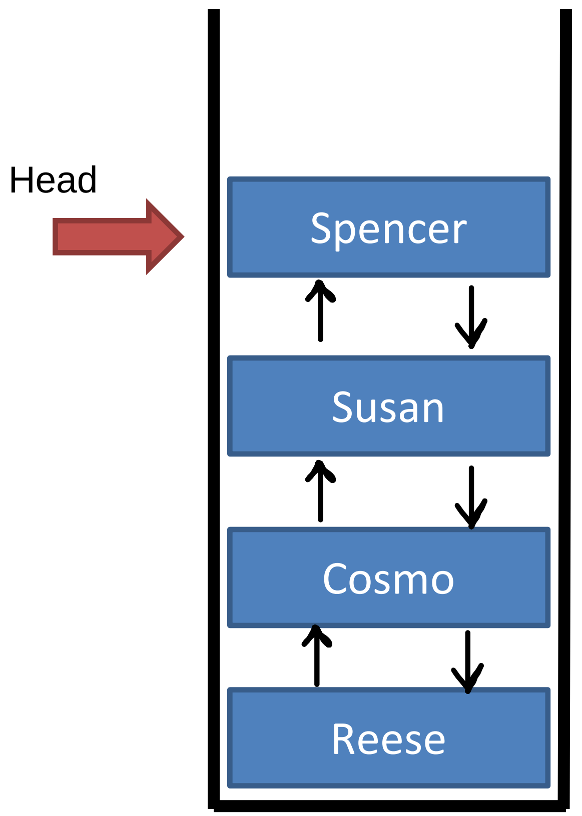
Whenever we remove an element from the stack (`pop()`), we always remove the head node of the linked list

The top of the stack will be the head of the linked list

Stack Implementation (Linked List)

To Do List:
- Push()
- Pop()
- Peek()
- IsEmpty()

We will import the Linked List Library (we will not write our own linked list class)



```
public void push(newElement){

    addToFront(newElement);
    size++

}
```

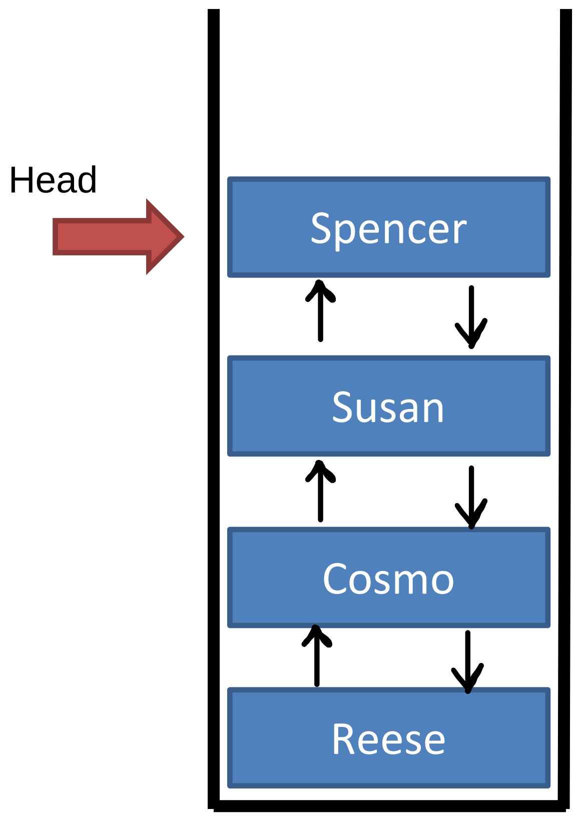
When we use a linked list, we are no longer restricted by a fixed size

Stack Implementation (Linked List)

To Do List:

- Push()
- Pop()
- Peek()
- IsEmpty()

We will import the Linked List Library (we will not write our own linked list class)



```
public void push(newElement){

    addToFront(newElement);
    size++
    top_of_stack = head

}

public void pop(){
     If size == 0:
        return
     Else:
        removeFront()
        size--
        top_of_stack = head

}
```

When we use a linked list, we are no longer restricted by a fixed size