CS 313
- Kangmei Yang
LinkedList

- Collection of elements implement List
- Each element may store anywhere in memory
- Each element comes with a reference and the element is linked one after another by these reference

```
head  ↓
e1    → e2    → e3    → e4    ↓
       ↑
tail
```
Node

- One element with a reference to next node

- class Node{
  int data;
  Node next;
}

- class Node<E>{
  E element;
  Node<E> next;
}

```
int data  next
```

```
element  next
```

```
  element
  ↓
```

Page 3
Node class

```java
 class Node<E>{
    private E element;
    private Node<E> next;
    public Node(E e, Node<E> n){
        element = e;
        next = n;
    }
    public E getElement(){
        return element;
    }
    public Node<E> getNext(){
        return next;
    }
    public void setElement(E e){
        element = e;
    }
    public void setNext(Node<E> n){
        next = n;
    }
}
```
LinkedList

- Store each data in a Node and link the Nodes

- Initially has no Nodes, create more Nodes as more data are added.

```java
public void add(E e){
    Node<E> nNode = new Node<>(e, null);
    Node<E> temp = head;
    while (temp.getNext() != null)
        temp = temp.getNext();
    temp.setNext(nNode);
}
```
public void add(E e){
    Node<E> nNode = new Node<>(e, null);
    Node<E> temp = head;
    while (temp.getNext() != null)
        temp = temp.getNext();
    if (head == null) head = nNode;
    else tail.setNext(nNode);
    tail = nNode;
}
public void add(E e){
    Node<E> nNode = new Node<>(e, null);
    tail.setNext(nNode);
    tail = nNode;
}
It’s efficient to add an element to the end of the list. What about deleting last element?
Doubly LinkedList

To make the delete and access the middle of the list more efficiently. We’d like to a doubly linkedlist. That is, the traverse forward and backward in either direction.
Doubly LinkedList

- header
  - null
  - trailer

prev null next
prev data next
prev null next
Loop

Two reference hop onto next node at two different speed, if there’s a loop, eventually they will meet.
Circularly Linkedlist

Has a reference to the last node. The first node is last.getNext()