Week 4 - Wednesday

# COMP 2100

# Last time

- What did we talk about last time?
- Array implementation of queues
- Started linked lists

# Questions?

# Project 1

Bitmap Manipulator

# Implementations

# Levels of flexibility

- Class protecting nodes implementation
- Generic class providing nodes with arbitrary type
- Generic class with the addition of iterators

# Singly linked list

- Node consists of data and a single next pointer
- Advantages: fast and easy to implement
- Disadvantages: forward movement only

**head**

| 23 | → | 47 | → | 58 | → X |
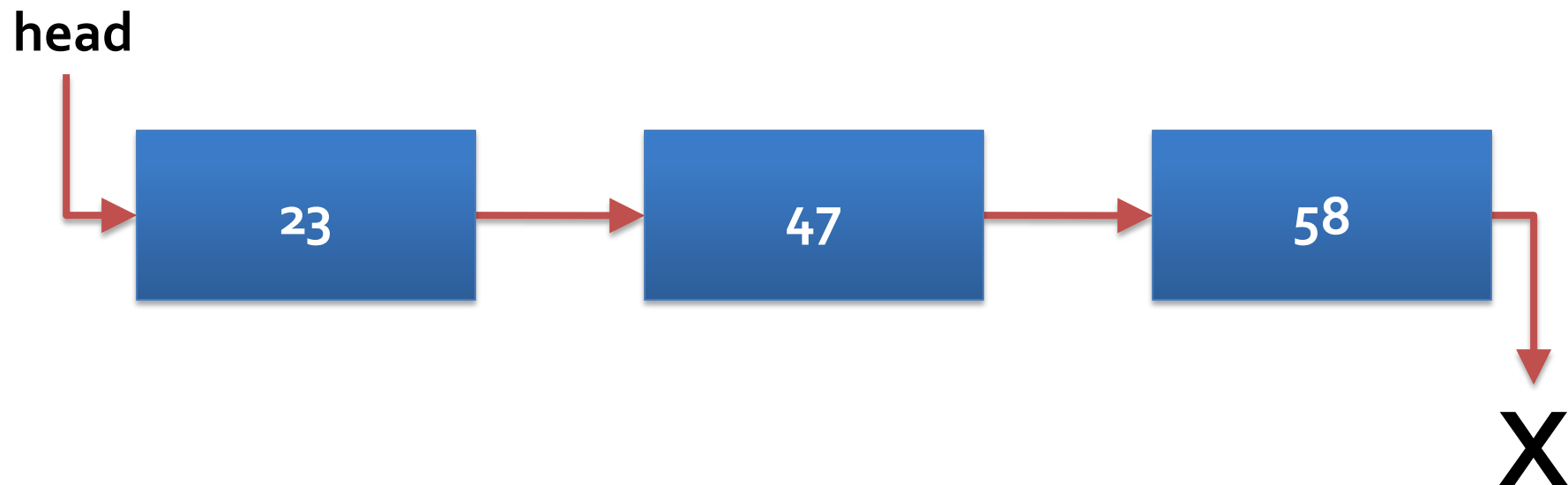
# Doubly linked list

- Node consists of data, a next pointer, and a previous pointer
- Advantages: bi-directional movement
- Disadvantages: slower, 4 pointers must change for every insert/delete
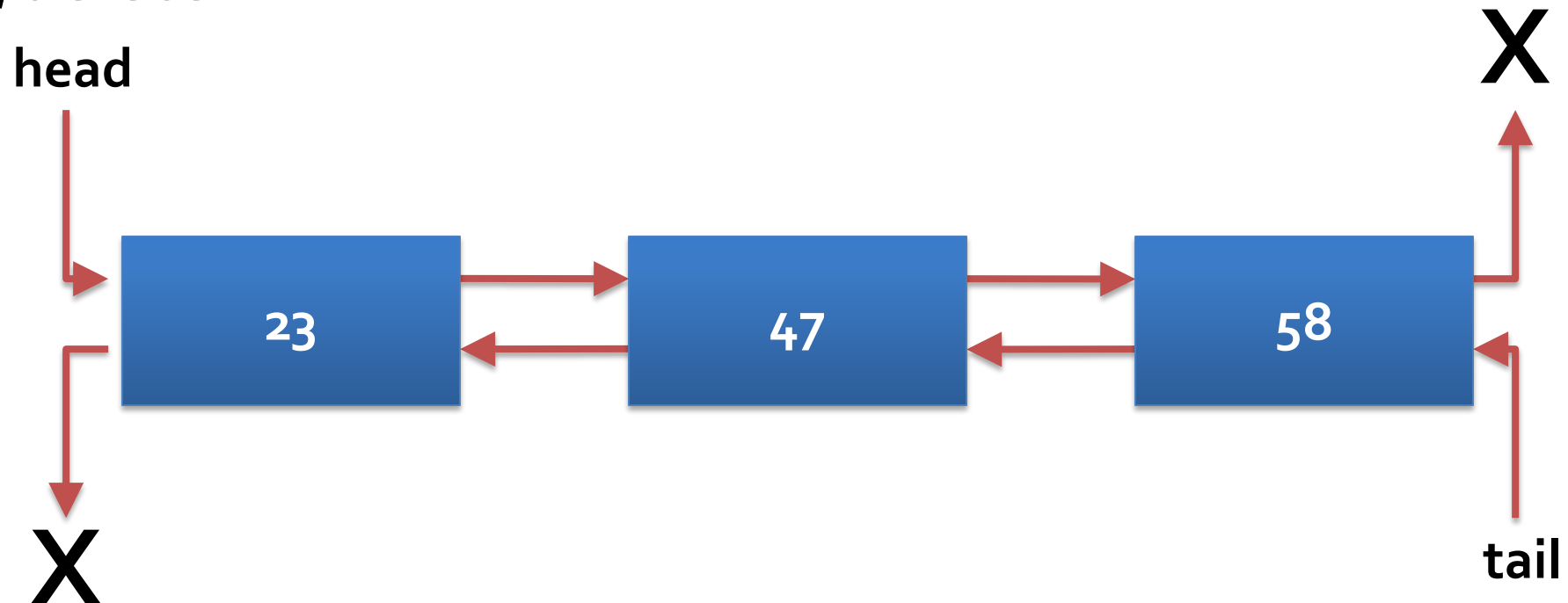
# Definition

- Let's try a simple definition for a generic doubly linked list:

```
public class LinkedList<T> {
    private static class Node<T> {
        public T data;
        public Node<T> next;
        public Node<T> previous;

    }

    private Node<T> head = null;
    private Node<T> tail = null;
    private int size = 0;
    …
}
```

# Add last

- Add to the end of the list

```java
public void add(T element) {


}
```

# Remove first

- Removes the first thing in the list and returns it

```
public T removeFirst() {



}
```

# Index Of

- Returns the index of the first occurrence of a specified element or -1 if this list does not contain the element

```java
public int indexOf(Object element) {

}
```

# Add at index *i*

- Inserts the specified element at the specified position in this list (shifting others down automatically)

```
public void add(int index, T element) {



}
```

# Quiz

# Upcoming

# Next time…

- Stack implementation with linked lists
- Queue implementation with linked lists
- Review for Exam 1

# Reminders

- Keep reading Chapter 3
- Keep working on Project 1
  - **Due this Friday, September 20 by midnight**
- **Exam 1 next Monday**