CS62: Fall 2025 | Lecture #7 (Doubly Linked Lists) worksheet | Prof. Li

1. Suppose x and t are references to different Nodes in a doubly linked list. What is the effect of the following code fragment?

```
t.prev = x;
t.next = x.next;
x.next.prev = t;
x.next = t;
```

2. What if instead the code was in a different order, i.e.:

```
x.next = t;
x.next.prev = t;
t.next = x.next;
t.prev = x;
```

3. Write the addLast method for DLLists. The addLast for SLLists is provided as reference, though you may not find it helpful. (Why? What method would be more helpful to reference?)

```
public void addLast(E element) {
      Node finger = head;
      while (finger.next != null) {
          finger = finger.next;
      }
      Node n = new Node();
      n.element = element;
      finger.next = n;
      size++;
  };
```

```
public void addLast(E element) {
    // Save old tail


    // Make a new node and assign it to tail. Fix pointers.









    // if first node to be added, adjust head to it.



    // else fix next pointer to tail



    // increase size


}
```

4. Fill in the blanks to implement remove at a specific index for DLLists.

```
public E remove(int index) {
        if (index >= size || index < 0) {
            throw new IndexOutOfBoundsException("Index out of bounds");
        }
        if (index == 0) {
            return ___________________
        } else if (index == size - 1) {
            return ___________________
        } else {
            Node previous = null;
            Node finger = head;
            // search for value indexed, keep track of previous
            while (index > 0) {
                previous = _________
                finger = ___________
                ___________ // what do we do with index?
            }
           // found the element to remove, change pointers
            previous.next = ____________;
            _____________ = previous;
            size--;
            return finger.element;
        }
    }
```

5. Check all the methods that are the same between singly linked lists and doubly linked lists (i.e., their implementation is the same, you can just copy/paste the SLL code and it will work for DLL).

☐ void clear() ☐ E get(int index) ☐ boolean isEmpty()

☐ E set(int index, E element) ☐ int size()