CS62: Fall 2025 | Lecture #6 (Singly Linked Lists) worksheet | Prof. Li

1. Suppose x is a reference to a Node and that node is not the last one on the singly linked list. What is the effect of the following code fragment?

```
x.next = x.next.next;
```

2. Suppose x and t are references to different Nodes in a singly linked list. What is the effect of the following code fragment? (Hint: draw it out)

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

3. Implement add at a specific index. For reference, here is add at the start of the list.

```
    public void add(E element) {
        // Save the old node
        Node oldHead = head;

        // Make a new node and assign it to head. Fix pointers.
        head = new Node();
        head.element = element;
        head.next = oldHead;

        //increment size
        size++;
    };

/**
 * Inserts the specified element at the specified index.
 *
 * @param index - the index to insert the element
 * @param element - the element to insert
 * @pre: 0<=index<=size()
 */
public void add(int index, E element) {
    if (index > size || index < 0){
        throw new IndexOutOfBoundsException("Index " + index + " out of bounds");
    }









}
```

4. Implement remove and return at a specific index. Hint: follow the pattern of Q3. For reference, here is remove from the start of the list.

```
public E remove() {
    Node temp = head;
    head = head.next;
    size--;
    return temp.element;
}
```

```
 /**
 * Retrieves and removes the element at the specified index.
 *
 * @param index
 *            the index of the element to be removed
 * @return the element previously at the specified index
 * @pre: 0<=index<size()
 */
public E remove(int index) {
 if (index >= size || index < 0){
     throw new IndexOutOfBoundsException("Index " + index + " out of bounds");
 }









}
```