Quiz 1 (Sept 20)

By taking this quiz, you agree to adhere to the honor code of the class.

Name: netid:

Write your name and netid on **both** sides of the paper. Write your solution **first on this side**. If space is not enough, write to the other side. You can ask for extra paper if necessary.

Name: netid:

Assume the type E to be integer. Implement `DoublyLinkedList<E> removeEven()` for DoublyLinkedList. This method should return a new doubly linked list that contains all the nodes with even elements. The original linked list should no longer contain even elements after the function call. You **cannot create new nodes** besides the sentinels. The order of elements must be preserved.
Example input:
Original: 3 ⇔ 5 ⇔ 10 ⇔ 8 ⇔ 1 ⇔ 2

Return list 10 ⇔ 8 ⇔ 2
The original linked list becomes 3 ⇔ 5 ⇔ 1

Signatures

```
public class DoublyLinkedList<E> implements PositionList<E>{
        private static class Node<E> implements Position<E>{
                private E element;
                private Node<E> prev;
                private Node<E> next;
                public Node(E e, Node<E> p, Node<E> n) {
                        element = e; prev = p; next = n;
                }
                public E getElement() { return element; }
                public void setElement(E element) { this.element = element; }
                public Node<E> getPrev() { return prev; }
                public Node<E> getNext() { return next; }
                public void setPrev(Node<E> p) { prev = p; }
                public void setNext(Node<E> n) { next = n; }
        }

        private Node<E> header;
        private Node<E> trailer;
        private int size = 0;

        public DoublyLinkedList( ) {
                header = new Node<>(null, null, null);
                trailer = new Node<>(null, header, null);
                header.setNext(trailer);
        }
```

Reference solution

```
public DoublyLinkedList<E> removeEven() {
    DoublyLinkedList<E> even = new DoublyLinkedList<E>();
    Node current = header.next;
    while(current != trailer) {

        if ((int)(current.element)%2==0) {
            Node temp = current;

            // Skip this one from the list
            current.prev.next = current.next;
            current.next.prev = current.prev;
            current = current.next;

            // Add to the evenlist
            temp.next = even.trailer;
            temp.prev = even.trailer.prev;
            even.trailer.prev = temp;
            temp.prev.next = temp;
        } else {
            current = current.next;
        }
    }

    return even;
}
```