Tyler Citrin

CSCI-343 Midterm Review

**I. Iterators**

Implement an iterator over a custom linkedlist class you made that **skips every other element**.

public Iterator<T> iterator() {

return new Iterator<T>() {

Node<T> currNode = head;

Node<T> previous = null;

public boolean hasNext() {

}

public T next() {

}

};

}

**Why do we use iterators?**

**II. Doubly Linked List** *(represented just by the root node here)*

**2.1 Draw a simple representation of a doubly linked list.**

**2.2 Now write code to determine if that linked list is circular**

public boolean isCircular(Node<T> list) {

}

**2.4** **Draw a simple representation of a doubly linked list, then the result of deleting the tail**

**2.5 Write code to delete tail of doubly linked list and then realigns it accordingly, then return the head**

public Node<T> deleteTail(Node<T> list) {

}

III. **Arrays**

3.1 Matrices: write a function that transposes a matrix in place, in the shortest time possible, then circle the numbers below that are checked.

[{1, 2, 3, 4},

{5, 6, 7, 8},

{9, 10, 11, 12},

{13, 14, 15, 16}]

3.2 Merge-sort is an algorithm that recursively halves, sorts, and merges an array.

For example, with an array [3, 1, 4, 2], the first split would yield [3, 1], [4, 2], and subsequently

[3], [1], [4], [2]. The next step would be to put it put it back together in sorted order.

[3, 1, 4, 2]

[3, 1] [4, 2]

[3] [1] [4] [2]

Draw the full recursive tree (this means it has a root at the top and bottom) of [38, 27, 43, 3, 60, 10].