CSE 214 Spring 2023

Recitation 2: Linked Lists

1. [5 Minutes] For the following problems, state the most efficient data structure(s) to use for the situation and explain why (Array, Singly Linked List, Doubly Linked List):
    a. Elements can frequently be accessed randomly

    b. Elements can be accessed sequentially both forwards and backwards

    c. Values can be inserted to the front of the structure

    d. Values can be accessed in reverse order from the end of the collection to the beginning

    e. The size of the collection has a fixed size to store data

2. [10 Minutes] In the following table, fill out the worst case time complexities for each operation. Assume sorted means from least to greatest and that you have access to both **head** and **tail** (unless otherwise stated):

	Unsorted Singly Linked List	Sorted Singly Linked List	Unsorted Doubly Linked List	Sorted Doubly Linked List
Traversing Values				
Inserting a node				
Deleting a node that you have access to				
Finding the minimum value				
Remove head				
Remove tail				
Insert new head				

Insert new tail				

3. [10 Minutes] Fill in the expressions for the following methods, which belong to a class SinglyLinkedList, which contains reference to the **head**, the **tail**, the **cursor**:

a.
```
/**
* Reverses the linked list
**/
public void reverse() {
        Node curr = this.head;
        Node prev = null;
        Node next = null;
        while (curr != null) {
                next = ___a___;
                ___b___;
                prev = curr;
                curr = ___c___;
        }
        this.head = ___d___;
}
```
a: _____ b: _____

c: _____ d: _____

a.
```
/**
* Removes the tail node and returns it
**/
public Node removeTail() {
        Node curr = this.head;
        if(curr ==___a___)
                return null; //The list is already empty
        Node prev = null;
        while (___b___ != null) {
                prev = curr;
                curr = ___c___;
        }
        this.tail = prev;
        if(___d___)
                prev.setNext(null);
        return ___e___;
```
a: _____ b: _____

c: _____ d: _____

e: _____

4. [10 Minutes] We wish to store a sequence of doubles using either an array or a singly-linked list of nodes with a head reference. Each node stores a data value and a reference to the next node. We know our sequence can contain up to 500 values. Assume that a double is 8 bytes and a memory reference is 4 bytes. You may ignore the references of the array and the head.

a. If we want to store 300 numbers in the sequence, which structure would be more memory efficient?

b. If we want to store 600 numbers in the sequence, which structure would be more memory efficient?

c. How many numbers could we store in the sequence such that neither structure is more efficient than the other?

d. Assuming the number of numbers in our sequence is N and that this number is known, find the time complexities of the following operations for both the array and the singly-linked list and explain:
   i. Find the first item in the sequence

   ii. Find the last item in the sequence

   iii. Insert a new head into the sequence

   iv. Insert a new tail into the sequence