


Name: [REDACTED] (as it will appear on official course roster)		
Email address: [REDACTED]	@umail.ucsb.edu	
Optional: name you wish to be called if different from name above.		
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone")		

1
h05
CS24 W19

h05: Chapter 9: Recursion

ready?	assigned	due	points
true	Wed 02/06 02:00PM	Mon 02/11 09:00AM	50

You may collaborate on this homework with AT MOST one person, an optional "homework buddy".

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE, OR IF APPLICABLE, SUBMITTED ON GRADESCOPE. There is NO MAKEUP for missed assignments; in place of that, we drop the lowest scores (if you have zeros, those are the lowest scores.)

Complete your reading of Chapter 9 (If you don't have a copy of the textbook yet, there is one on reserve at the library under "COMP000-STAFF - Permanent Reserve").

Please:

- No Staples.
- No Paperclips.
- No folded down corners.

1. (10 pts) Consider the following definition of a doubly linked-list.

```
class LinkedList{
public:
    LinkedList():head(0), tail(0){}
    ~LinkedList();
    void reverse(); //reverses the order of elements in the linked list
    void insert(int value);
private:
    struct Node{
        int data;
        Node* next;
        Node* prev;
    };
    Node* head;
    Node* tail;
    //Add your helper function here that recursively reverses the order of elements in the linked list.
    void reverseHelper(Node* n, Node* t);
};
```

Write the declaration of a helper function in the class provided above that recursively reverses the order of elements in the linked list. This function will be used by the reverse() method. The implementation of the function should be provided on the next page.

9 (20 pts) Consider the doubly linked list in the previous question. Implement the reverse method that uses a helper function to recursively reverse the order of elements in a linked list. You must implement the helper function as well.

```
void LinkedList::reverseHelper(Node* h, Node* t) {
    if (h == t)
        return;

    int tmp = t->data;
    t->data = h->data;
    h->data = tmp;
    if (h->next == t) {
        return;
    } else {
        reverseHelper(h->next, t->prev);
    }
}
```

2

h05

CS24 W19

```
void LinkedList::reverse() {
    reverseHelper(head, tail);
}
```