

# Copy by reference, Pass by Value, Instantiating Arrays

Justin, Nicki, Christine, Kisha, Sreesha

## Question 1

Fill in the blanks of the code to remove a doubly linked list's node. Remember that you can access a node's previous and next elements.

```
public class Noads {  
    public void remove(Node n){  
        node._____ = node._____;  
        node._____ = node._____;  
        node._____ = null;  
        node._____ = null;  
    }  
}  
  
node.previous.next = node.next  
node.next.previous = node.previous  
node.previous = null  
node.next = null
```

## Question 2

Using Professor Hug's code for SentinelSList, write the method called *reverse* which reverses the nodes in a SentinelSList without changing the position of the sentinel.

```
public void reverse() {
    int sentinel = front.item;
    intNode reversed = null;
    intNode temp = front;

    while (temp != null){
        IntNode tail = temp.next;
        temp.next = reversed;
        reversed = temp;
        temp = tail;
    }
    reversed = new IntNode(sentinel, reversed);
    front = reversed;
}
```

### Question 3

Using the given SList Class, implement middle method which returns the object in the middle of the list. (Hint: Use two pointers instead of one).

```
public class SList {
    private int head;
    private SList next = null;

    public SList(int head, SNode k) {
        this.head = head;
        this.next = k;
    }

    public SList() {
        this(null);
    }

    public static Object middle (Slist list) {
        SListNode slow = list.head;
        SListNode fast = list.head;
        while (fast.next != null && fast.next.next != null) {
            slow = slow.next;
            fast = fast.next.next;
        }
        return slow;
    }
}
```

## Question 4

Using the given SList Class, as Question 3, implement remove method which removes objects in every odd number index. *For example*, given SList a = [1, 2, 3, 4, 5], remove function should mutate a to [2, 4].

```
public class SList {
    private int head;
    private SList next;

    public SList(int h, SNode k) {
        this.head = head;
        this.next = k;
    }

    public SList() {
        this(null);
    }

    public static void remove (SList a) {
        SListNode pointer = a;
        int counter = 0;
        while (pointer.next != null && pointer != null ) {
            if(counter % 2 == 0) {
                pointer.next = pointer.next.next;
            }
            pointer = pointer.next;
            counter = counter + 1;
        }
        return a;
    }
}
```

## 1 Question 5

Question 5 Description