Homework 2

Qinlang Chen qc2176

1.

(a)

```
public int size()
{
    Node current = header;
    int size = 0;
    while(current != null){
        size ++;
        current = current.next;
    }
    return size;

}
```

(b)

```
public void print()
{
    Node current = header;
    String result = (String) current.data;
    current = current.next;
    while(current != null){
        result += " " + (String) current.data;
        current = current.next;
    }
    System.out.format("LinkedList Contains: %s\n", result);
}
```

(c)

```
public Boolean contain(Object x)
{
    Node current = header;
    Boolean result = false;
    while(current != null){
        if(current.data == x){
            result = true;
            break;
        }
        current = current.next;
    }
```

```
    return result;
}
```

(d)

```
public Object add(Object x)
{
    Node current = header;
    Node result = new Node(x, null);
    if(header == null){;
        header = result;
        return result.data;
    }
    if(this.contain(x))
        return null;
    while(current.next != null)
        current = current.next;
    current.next = result;
    return result.data;
}
```

(e)

```
public Object remove(Object x)
{
    Node current = header;
    Node next;
    if(header.data == x){
        if(header.next == null){
            header = null;
            return x;
        }
        header.next = header.next.next;
        return x;
    }
    while(current.next != null){
        next = current.next;
        if(next.data == x){
            current.next = next.next;
            return x;
        }
        current = current.next;
    }
    return null;
}
```

(f)

```
public void reverse()
{
    if (header == null || header.next == null)
        return;
    Node a = header;
    Node b = a.next;
    if (b.next == null){
        b.next = a;
        a.next = null;
        header = b;
        return;
    }
    Node c = b.next;
    b.next = a;
    a.next = null;
    a = b;
    b = c;
    while(b.next != null){
        c = b.next;
        b.next = a;
        a = b;
        b = c;
    }
    b.next = a;
    header = b;
}
```

(g)

```
public LinkedList intersection(LinkedList L1, LinkedList L2)
{
    LinkedList L3 = new LinkedList();
    Node current1 = L1.header;
    Node current2 = new Node();
    while(current1 != null){
        current2 = L2.header;
        while(current2 != null){
            if(current1 == current2){
                L3.add(current1.data);
                break;
            }
            current2 = current2.next;
        }
```

```
            current1 = current1.next;
        }
        return L3;
    }
```