```
class Nodee {

    private Object data;

    private Nodee next;



    //constructor

    public Nodee(Object newData)

    {

        data = newData;

        next = null;

    }



    //The node pointed to by next is returned

    public Nodee getNext()

    {

        return(next);

    }



    //The node pointed to by next is changed to newNode

    public void setNext(Nodee newNode)

    {

        next = newNode;

    }
```

I recycled node from my last mastery, and changed the variables a little to make them more distinct.

```
package stacklist;


public class linkedlistt

{

    private Nodee head;


    //constructor

    public linkedlistt()

    {

        head = null;

    }


    public void setempty()
    {
        head=null;
    }


    public Object gethead()
    {
        return(head.getData());
    }


    //Adds a node to the linked list.

    public void addAtFront(Object str)

    {

        Nodee newNode = new Nodee(str);

        newNode.setNext(head);

        head = newNode;

    }




    //Adds a node to the end of the linked list.

    public void addAtEnd(Object str)
```

```
//Adds a node to the end of the linked list.
public void addAtEnd(Object str)

{

    Nodee current = head;

    Nodee newNode = new Nodee(str);


    while (current.getNext() != null)

    {

        current = current.getNext();

    }

    current.setNext(newNode);

}
```

```
//Deletes a node in the linked list.

public Object remove()

{

    Nodee current = head;
    head =head.getNext();
    return(current.getData());
}

public int size()
{
    Nodee current = head;
    int listnum = 0;

    if (current != null)
    {
        listnum +=1;

        while (current.getNext() != null )
        {
            listnum +=1;

            current = current.getNext();
        }
    }
```

I started by copying linked-list from another project

```
private linkedlistt data;
private int top;

public stacklist()
{
    data = new linkedlistt();
}

public Object top()
{
    return (data.gethead());
}



public Object pop()
{

data.remove();

return data;
}

public void push(Object item)
{
    data.addAtFront(item);
}

public boolean is_empty()
{
    if (data.size()==0)
    {
        return true;
    }
    else
    {
        return false;
    }


}

public void make_empty()
{
    data.setempty();
}


public int size()
{
    int num;
    num = data.size();
    return (num);
}
```

I started by copying stack 2 then modifying it to fit with linked list and node.

```
public class stackListTest {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        String hi = "hi";

        stacklist s2 = new stacklist();


        s2.push("red");
        s2.push("yellow");
        s2.push("green");
        s2.push("red");
        s2.push("reb");

        System.out.println("top of stack s2 is: " + s2.top());
        System.out.println("how many items r in stack 2:  " + s2.size());

        s2.pop(); //removes the top one

        System.out.println("top of stack s2 is: " + s2.top());
        System.out.println("how many items r in stack 2:  " + s2.size());

        s2.make_empty();



    }

}
```

Copied test from stack 2