

```
1  /*****
2  * Dalton Nofs
3  * Login ID: nofs5491
4  * CS-102, Summer 2017
5  * Programming Assignment 4
6  * Node class: node object for linkedLists
7  *****/
8  public class Node<T>
9  {
10     T data;    // Data to be stored in Node
11     Node<T> next;    // Next node ptr
12     Node<T> previous;    // Previous node ptr
13
14     /*****
15     * Method: Node()
16     * Purpose: default constructor for node obj
17     *
18     * Parameters:          N/A
19     * Returns: void:      N/A
20     *****/
21     public Node()
22     {
23         data = null;
24         next = null;
25         previous = null;
26     }
27
28     /*****
29     * Method: getData()
30     * Purpose: get previous node or next node
31     *
32     * Parameters:          N/A
33     * Returns: Object:     data type stored in node
34     *****/
35     public T getData()
36     {
37         return data;
38     }
39
40     /*****
41     * Method: setData()
42     * Purpose: get previous node or next node
43     *
44     * Parameters: T:       Data to be stored in node
45     * Returns: void:      N/A
46     *****/
47     public void setData(T data)
48     {
49         this.data = data;
50     }
51
52     /*****
53     * Method: getNext()/getPrevious()
54     * Purpose: get previous node or next node
55     *
56     * Parameters:          N/A
57     * Returns: Node:       Node pointing to next or prev
58     *****/
59     public Node<T> getNext()
60     {
61         return next;
62     }
63     public Node<T> getPrevious()
64     {
65         return previous;
66     }
67 }
```

```
68  /*****
69  * Method: setNext()/setPrevious()
70  * Purpose: Set the next or previous node
71  *
72  * Parameters: Node:      Next/previous node
73  * Returns: Void:        N/A
74  *****/
75  public void setNext(Node<T> next)
76  {
77      this.next = next;
78  }
79  public void setPrevious(Node<T> previous)
80  {
81      this.previous = previous;
82  }
83
84 }
```