CSCI 3200 Spring 2024
Optional Group Project
Due date: April 26, 2025 @ 11:59 pm
100 Points

Directions:

1. You can form teams of 2-3 individuals to work on this project.
2. Write the body of the methods using the method header that is provided in this document. Note: if you re-write the method header points will be subtracted.
3. This project is based on a binary search tree using linked list structures. Create a class named BinarySearchTree_LinkedList.java.
4. Download the Node.java and BinarySearchTree_LinkedList files from D2L.
5. Add BinarySearchTree_LinkedList class to the project:

```
public class BinarySearchTree_LinkedList {
    protected Node<Integer> root;

    public BinarySearchTree_LinkedList(int element) {
            // create a node using the parameter "element"
    }

    public boolean isRoot(int value) {
            // checks if the value is the root node
    }

    public Node<Integer> insert(Node<Integer> rt, Node<Integer> add) {
            // insert a node in the left or right subtree.
            // use recursion to ensure the node is placed in the correct
            // position to //maintain the characteristics of a binary search tree
    }

    public Node<Integer> findmax(Node<Integer> node) {
            // searches the left subtree to
            // return the maximum node
    }

    public Node<Integer> findmin(Node<Integer> node) {
            // searches the right subtree to return the minimum node
```

```
    }

    public Node<Integer> remove(Node<Integer> temp, int value) {
        // remove a node from the tree. But it should maintain the
        // characteristics of a binary search tree. Use findmin or findmax method
    }

    public int parent(Node<Integer> rt, Node<Integer> add) {
        // search the left subtree and right subtree for the node
        // first starting from the root node n(rt). When found return the value of its
        // parent. //Return -1 if the value is not in the tree or the node is root.
    }

    public boolean searchNode(Node<Integer> rt, int value) {
        // search the left subtree or right subtree for a value. Note search
        // always start from the root node and returns true if the value is found
        // or false if not found
    }

    public LinkedList<Node<Integer>> children(Node<Integer> root, int value,
                                              LinkedList<Node<Integer>> myList) {
        // checks if the parent node is in the binary tree starting from the root
        // node. If true, then returns a LinkedList with the child nodes).
    }

}
```

6. Complete the method body for each method by following the instructions given in each method as a comment.

Expected Output:

The minimum value in the binary tree is: 1
The maximum value in the binary tree is: 14
Found the node in the tree!
The child node(s) is/are: 1
The parent is 5

Note: take a screenshot of your program running and submit it to D2L with the BinarySearchTree_LinkedList.java file. Assumption: all the values in the nodes of this tree are unique. The method headers outlined above should be used in the creation of the methods. Do not modify them.