```java
package binarytree;

import java.util.*;
import java.io.*;
class Node34 {
    int data;
    Node34 left, right, nextRight;

    Node34(int item)
    {
        data = item;
        left = right = nextRight = null;
    }
}

  class BinaryTree62 {
    Node34 root;
    void connect(Node34 p)
    {
        // initialize queue to hold Node34s at same level
        Queue<Node34> q = new LinkedList<>();

        q.add(root); // adding Node34s to tehe queue

        Node34 temp = null; // initializing prev to null
        while (!q.isEmpty()) {
            int n = q.size();
            for (int i = 0; i < n; i++) {
                Node34 prev = temp;
                temp = q.poll();

                // i > 0 because when i is 0 prev points
                // the last Node34 of previous level,
                // so we skip it
                if (i > 0)
                    prev.nextRight = temp;

                if (temp.left != null)
                    q.add(temp.left);

                if (temp.right != null)
                    q.add(temp.right);
            }

            // pointing last Node34 of the nth level to null
            temp.nextRight = null;
        }
    }

    // Driver program to test above functions
    public static void main(String args[])
    {
        BinaryTree62 tree = new BinaryTree62();

        /* Constructed binary tree is
            10
            / \
        8    2
        /
        3
        */
        tree.root = new Node34(10);
        tree.root.left = new Node34(8);
        tree.root.right = new Node34(2);
        tree.root.left.left = new Node34(3);
```

```
        // Populates nextRight pointer in all Node34s
        tree.connect(tree.root);

        // Let us check the values of nextRight pointers
        System.out.println("Following are populated nextRight pointers in "
                + "the tree"
                + "(-1 is printed if there is no nextRight)");
        int a = tree.root.nextRight != null ? tree.root.nextRight.data : -1;
        System.out.println("nextRight of " + tree.root.data + " is "
                + a);
        int b = tree.root.left.nextRight != null ? tree.root.left.nextRight.data : -1;
        System.out.println("nextRight of " + tree.root.left.data + " is "
                + b);
        int c = tree.root.right.nextRight != null ? tree.root.right.nextRight.data : -1;
        System.out.println("nextRight of " + tree.root.right.data + " is "
                + c);
        int d = tree.root.left.left.nextRight != null ? tree.root.left.left.nextRight.data : -1;
        System.out.println("nextRight of " + tree.root.left.left.data + " is "
                + d);
    }
}
// This code has been contributed by Rahul Shakya
```