Objective(s):

a. To be able to implement binary-search-tree delete(BNode n, int d) method
b. To be able to implement binary tree rotation method

Task 1: Create MyBST_XXYYYY.java which extends MyBST_Basic_XXYYYY.

- implement BNode getRoot() **in** MyBST_Basic_XXYYYY since we will send the tree to a static method (You'll call it in /* your code 1 */).
- Implement findMin(BNode n) and findMax(BNode n)

```
package Lab10a.pack;

public class MyBST_XXYYYY extends MyBST_Basic {
  public MyBST_XXYYYY() { super() };
  public MyBST_XXYYYY(Integer[] input) {
      super(input);
  }
  public BNode getRoot() {
      /* your code 1 */;
  }
  public BNode findMin(BNode node) {
      if (node == null) throw new IllegalArgumentException("Empty Tree");
      while (node.left != null) {
          node = node.left;
      }
      return node;
  }
  public BNode findMax(BNode node) {
      if (node == null) throw new IllegalArgumentException("Empty Tree");
      while (/* your code 2 */) {
          node = node.right;
      }
      return node;
  }
```

```
package Lab10a;
  …
  public static void demo1() {
    System.out.println("---- find min & max ----");
    Integer data[] = {45, 12, 30, 55, 31, 64, 59};
    MyBST_XXYYYY bst = new MyBST_XXYYYY(data);

    System.out.println(bst.findMin(bst.getRoot())); // null <- 12 -> 30
    System.out.println(bst.findMax(bst.getRoot())); // 59 <- 64 -> null

    bst.insert(70);
    System.out.println(bst.findMax(bst.getRoot())); // null <- 70 -> null
  }
```

Task 2: implement `delete(BNode n, int d)`

```
public static void demo2() {
    System.out.println("---- delete ----");
    Integer data[] = {15, 34, 20, 10, 18, 16, 12, 8};
    MyBST_XXYYYY bst = new MyBST_XXYYYY(data);
    System.out.println(bst); // 8 10 12 15 16 18 20 34
    System.out.println(bst.search(10)); // 8 <- 10 -> 12
    System.out.println(bst.search(15)+"\n"); // 10 <- 15 -> 34

    bst.delete(bst.getRoot(), 34);
    bst.delete(bst.getRoot(), 10);
    bst.delete(bst.getRoot(), 100);
    System.out.println(bst); // 8 12 15 16 18 20
    System.out.println(bst.search(10)); // null
    System.out.println(bst.search(15)); // 8 <- 15 -> 20
    System.out.println(bst.search(20)); // 18 <- 20 -> null
}
```

Task 3: BST preserve order of the data on a non-linear structure. Yet, it is not hard to find them. Complete BNode inorderPredecessor (int d) to find predecessor of node with value d and complete BNode inorderSuccessor(int d) to find successor of node with value d.

```
public static void demo3() {
  System.out.println("---- find predecessor & successor ----");
  Integer data[] = {32, 46, 18, null, 43, 25, 31, 13};
  MyBST_XXYYYY bst = new MyBST_XXYYYY(data);

  System.out.println(bst.inorderPredecessor(31)); // null <- 25 -> 31
  System.out.println(bst.inorderSuccessor( 43)); // 43 <- 46 -> null
  System.out.println(bst.inorderPredecessor( 13)); // null
}
```

Task 4: Complete BNode kthSmallest(BNode n, int d) to return the k^{th} smallest value on the tree.

(hint: keep pushing root of left subtree so that you can go to right subtree if need.)

```
public static void demo4() {
  System.out.println("---- kth smallest ----");
  Integer data[] = {1, null, 4, 10, 5, 7};
  MyBST_XXYYYY bst = new MyBST(data);

  System.out.println(bst.kthSmallest(bst.getRoot(), 3)); // 5
  System.out.println(bst.kthSmallest(bst.getRoot(), 5)); // 10
}
```

Task 5: Complete

- static void leftRotate(MyBST_XXYYYY bst, BNode node)
- static void rightRotate(MyBST_XXYYYY bst, BNode node) // static void rightLeftRotate() is given

```
public static void demo5() {
    System.out.println("---- rotate ----");
    Integer[] data1 = {4,5,6,9};
    MyBST bst1 = new MyBST(data1);
    System.out.println(bst1.search(5)); // null <- 5 -> 6
    System.out.println(bst1.search(6)); // null <- 6 -> 9
    System.out.println(bst1.search(9)); // null <- 9 -> null

    MyBST.leftRotate(bst1, bst1.search(5));
    System.out.println(bst1.search(5)); // null <- 5 -> null
    System.out.println(bst1.search(6)); // 5 <- 6 -> 9
    System.out.println(bst1.search(9) + "\n"); // null <- 9 -> null

    Integer[] data2 = {2,4,7,5,6};
    MyBST bst2 = new MyBST(data2);
    System.out.println(bst2.search(4)); // null <- 4 -> 7
    System.out.println(bst2.search(7)); // 5 <- 7 -> null
    System.out.println(bst2.search(5)); // null <- 5 -> 6

    MyBST.rightLeftRotate(bst2, bst2.search(4));
    System.out.println(bst2.search(4)); // null <- 4 -> null
    System.out.println(bst2.search(7)); // 6 <- 7 -> null
    System.out.println(bst2.search(5)); // 4 <- 5 -> 7
}
```

Submission: Your **MyBST_XXYYYY.java** where. XX are the first two digit of your student id and YYYY are the last four.

Due date: TBA