Solution 25-11-23

449. Serialize and Deserialize BST

```
import java.util.Arrays;
import java.util.LinkedList;
import java.util.Queue;

class TreeNode {
  int val;
  TreeNode left;
  TreeNode right;

  TreeNode(int x) {
    val = x;
  }
}

public class BSTSerialization {
  // Serialize a BST to a string
  public String serialize(TreeNode root) {
    StringBuilder sb = new StringBuilder();
    serializeHelper(root, sb);
    return sb.toString();
  }

  private void serializeHelper(TreeNode node, StringBuilder sb) {
    if (node == null) {
      sb.append("null").append(",");
    } else {
      sb.append(node.val).append(",");
      serializeHelper(node.left, sb);
```

```
    serializeHelper(node.right, sb);
  }
}

// Deserialize a string to a BST
public TreeNode deserialize(String data) {
  Queue<String> nodes = new LinkedList<>(Arrays.asList(data.split(",")));
  return deserializeHelper(nodes);
}

private TreeNode deserializeHelper(Queue<String> nodes) {
  String val = nodes.poll();
  if (val.equals("null")) {
    return null;
  } else {
    TreeNode node = new TreeNode(Integer.parseInt(val));
    node.left = deserializeHelper(nodes);
    node.right = deserializeHelper(nodes);
    return node;
  }
}

// Example usage
public static void main(String[] args) {
  BSTSerialization bstSerialization = new BSTSerialization();

  // Example tree
  TreeNode root = new TreeNode(5);
  root.left = new TreeNode(3);
  root.right = new TreeNode(8);
  root.left.left = new TreeNode(2);
```

```
    root.left.right = new TreeNode(4);
    root.right.left = new TreeNode(6);
    root.right.right = new TreeNode(9);

    // Serialize the tree
    String serialized = bstSerialization.serialize(root);
    System.out.println("Serialized tree: " + serialized);

    // Deserialize the tree
    TreeNode deserialized = bstSerialization.deserialize(serialized);

    // You can perform operations on the deserialized tree as needed
  }
}
```

Output

```
PS C:\Users\Ajeet\Desktop\java> javac BSTSerialization.java
PS C:\Users\Ajeet\Desktop\java> java BSTSerialization
Serialized tree: 5,3,2,null,null,4,null,null,8,6,null,null,9,null,null,
```

```
import java.util.*;

public class IntersectionOfArrays {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    // Input for the first array
    System.out.print("Enter the size of the first array: ");
    int n = scanner.nextInt();
    int[] arr1 = new int[n];
    System.out.println("Enter elements of the first array:");
```

```
  for (int i = 0; i < n; i++) {
    arr1[i] = scanner.nextInt();
  }

  // Input for the second array
  System.out.print("Enter the size of the second array: ");
  int m = scanner.nextInt();
  int[] arr2 = new int[m];
  System.out.println("Enter elements of the second array:");
  for (int i = 0; i < m; i++) {
    arr2[i] = scanner.nextInt();
  }

  // Finding the intersection of the arrays
  List<Integer> intersection = findIntersection(arr1, arr2);

  // Displaying the intersection
  System.out.println("Intersection of the two arrays: " + intersection);

  // Closing the scanner
  scanner.close();
}

// Function to find the intersection of two arrays
private static List<Integer> findIntersection(int[] arr1, int[] arr2) {
  List<Integer> result = new ArrayList<>();
  Set<Integer> set1 = new HashSet<>();

  // Add elements of arr1 to set1
  for (int num : arr1) {
    set1.add(num);
```

```
    }

    // Check elements of arr2 against set1
    for (int num : arr2) {
      if (set1.contains(num)) {
        result.add(num);
        set1.remove(num); // Avoid duplicates in the result
      }
    }

    return result;
  }
}
```

Output

```
PS C:\Users\Ajeet\Desktop\java> javac IntersectionOfArrays.java
PS C:\Users\Ajeet\Desktop\java> java IntersectionOfArrays
Enter the size of the first array: 5
Enter elements of the first array:
1 2 3 4 5
Enter the size of the second array: 4
Enter elements of the second array:
5 6 7 8
Intersection of the two arrays: [5]
```