Solution 1

```
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

public class Program {
  // O(d) time | O(1) space - where d is the depth (height) of the ancestral tree
  public static AncestralTree GetYoungestCommonAncestor(
    AncestralTree topAncestor,
    AncestralTree descendantOne,
    AncestralTree descendantTwo
    ) {
    int depthOne = getDescendantDepth(descendantOne, topAncestor);
    int depthTwo = getDescendantDepth(descendantTwo, topAncestor);
    if (depthOne > depthTwo) {
      return backtrackAncestralTree(descendantOne, descendantTwo,
                depthOne - depthTwo);
    } else {
      return backtrackAncestralTree(descendantTwo, descendantOne,
                depthTwo - depthOne);
    }
  }

  public static int getDescendantDepth(AncestralTree descendant, AncestralTree to
    int depth = 0;
    while (descendant != topAncestor) {
      depth++;
      descendant = descendant.ancestor;
    }
    return depth;
  }

  public static AncestralTree backtrackAncestralTree(
    AncestralTree lowerDescendant,
    AncestralTree higherDescendant,
    int diff
    ) {
    while (diff > 0) {
      lowerDescendant = lowerDescendant.ancestor;
      diff--;
    }
    while (lowerDescendant != higherDescendant) {
      lowerDescendant = lowerDescendant.ancestor;
      higherDescendant = higherDescendant.ancestor;
    }
    return lowerDescendant;
  }

  public class AncestralTree {
    public char name;
    public AncestralTree ancestor;

    public AncestralTree(char name) {
      this.name = name;
      this.ancestor = null;
    }

    // This method is for testing only.
    public void AddAsAncestor(AncestralTree[] descendants) {
      foreach (AncestralTree descendant in descendants) {
        descendant.ancestor = this;
      }
    }
  }
}

```

Solution 1 | Solution 2 | Solution 3

```
public class Program {
  public static AncestralTree GetYoungestCommonAncestor(
    AncestralTree topAncestor,
    AncestralTree descendantOne,
    AncestralTree descendantTwo
    ) {
    // Write your code here.
    return null;
  }

  public class AncestralTree {
    public char name;
    public AncestralTree ancestor;

    public AncestralTree(char name) {
      this.name = name;
      this.ancestor = null;
    }

    // This method is for testing only.
    public void AddAsAncestor(AncestralTree[] descendants) {
      foreach (AncestralTree descendant in descendants) {
        descendant.ancestor = this;
      }
    }
  }
}

```

Run or submit code when you're ready.