AlgoExpert Quad Layout C++ 12px Sublime Monokai 00:00:0

Our Solution(s) Run Code

Solution 1

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

#include <vector>
using namespace std;

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

  AncestralTree(char name) {
    this->name = name;
    this->ancestor = NULL;
  }

  void addAsAncestor(vector<AncestralTree *> descendants);
};

int getDescendantDepth(AncestralTree *descendant, AncestralTree *topAncestor);
AncestralTree *backtrackAncestralTree(AncestralTree *lowerDescendant,
                                      AncestralTree *higherDescendant,
                                      int diff);

// O(d) time | O(1) space - where d is the depth (height) of the ancestral tree
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);
  }
}

int getDescendantDepth(AncestralTree *descendant, AncestralTree *topAncestor) {
  int depth = 0;
  while (descendant != topAncestor) {
    depth++;
    descendant = descendant->ancestor;
  }
  return depth;
}

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;
}
```

Your Solutions Run Code

Solution 1 **Solution 2** **Solution 3**

```cpp
#include <vector>
using namespace std;

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

  AncestralTree(char name) {
    this->name = name;
    this->ancestor = NULL;
  }

  void addAsAncestor(vector<AncestralTree *> descendants);
};

AncestralTree *getYoungestCommonAncestor(AncestralTree *topAncestor,
                                         AncestralTree *descendantOne,
                                         AncestralTree *descendantTwo) {
  // Write your code here.
  return NULL;
}
```

Custom Output **Raw Output** Submit Code

Run or submit code when you're ready.