Our Solution(s) Run Code

Solution 1

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

#include <vector>
#include <deque>
using namespace std;

class Node {
public:
  string name;
  vector<Node *> children;

  Node(string name) { this->name = name; }

  // O(v + e) time | O(v) space
  vector<string> breadthFirstSearch(vector<string> *array) {
    deque<Node *> queue{this};
    while (!queue.empty()) {
      Node current = *queue.front();
      queue.pop_front();
      array->push_back(current.name);
      for (int i = 0; i < current.children.size(); i++) {
        queue.push_back(current.children[i]);
      }
    }
    return *array;
  }

  Node *addChild(string name) {
    Node *child = new Node(name);
    children.push_back(child);
    return this;
  }
};
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

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

// Do not edit the class below except
// for the breadthFirstSearch method.
// Feel free to add new properties
// and methods to the class.
class Node {
public:
  string name;
  vector<Node *> children;

  Node(string str) { name = str; }

  vector<string> breadthFirstSearch(vector<string> *array) {
    // Write your code here.
    return {};
  }

  Node *addChild(string name) {
    Node *child = new Node(name);
    children.push_back(child);
    return this;
  }
};

```

Our Tests

Custom Output Submit Code

```
public class
    void Run() {

        RunTest("Test Case 1", () => {
            Node head1 = 
            head1.addChild('A');
            head1.addChild('B').addChild('C');
            head1.children[0].addChild('D');
            var result = Dfs(head1, 'A', 'B', 'C', 'D');
            var expected = new[] {
            Assert.IsTrue(result.SequenceEqual(expected));
        });

        RunTest("Test Case 2", () => {
            Node head2 = 
            head2.addChild('A').addChild('B').addChild('C').addChild('D').addChild('E');
            head2.children[2].addChild('F');
            var result = Dfs(head2, 'A', 'B', 'C', 'D', 'E', 'F');
            var expected = new[] {
            Assert.IsTrue(result.SequenceEqual(expected));
        });

```

Run or submit code when you're ready.