Our Solution(s) Run Code

Solution 1

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

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

class MinMaxStack {
public:
  vector<unordered_map<string, int>> minMaxStack = {};
  vector<int> stack = {};

  // O(1) time | O(1) space
  int peek() { return stack[stack.size() - 1]; }

  // O(1) time | O(1) space
  int pop() {
    minMaxStack.pop_back();
    int result = stack[stack.size() - 1];
    stack.pop_back();
    return result;
  }

  // O(1) time | O(1) space
  void push(int number) {
    unordered_map<string, int> newMinMax = {{"min", number}, {"m
    if (minMaxStack.size()) {
      unordered_map<string, int> lastMinMax =
          minMaxStack[minMaxStack.size() - 1];
      newMinMax["min"] = min(lastMinMax["min"], number);
      newMinMax["max"] = max(lastMinMax["max"], number);
    }
    minMaxStack.push_back(newMinMax);
    stack.push_back(number);
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```cpp
using namespace std;

// Feel free to add new properties and methods to the class.
class MinMaxStack {
public:
  int peek() {
    // Write your code here.
    return -1;
  }

  int pop() {
    // Write your code here.
    return -1;
  }

  void push(int number) {
    // Write your code here.
  }

  int getMin() {
    // Write your code here.
    return -1;
  }

  int getMax() {
    // Write your code here.
    return -1;
  }
};

```

Our Tests

Custom Output Submit Code

```
  }
    assert(stack.getMax() == max);
    assert(stack.getMax() == max);
    assert(stack.pop() == popValue);
  }

class Program1test : public TestSuite {
public:
  void Test() {

    RunTest("Test Case 4", []() {
      MinMaxStack stack;
      stack.push(3);
      testMinMaxPush(3, 3, 3, stack);
      stack.push(7);
      testMinMaxPush(3, 7, 7, stack);
      stack.push(5);
      testMinMaxPush(3, 7, 5, stack);
      stack.push(9);
      testMinMaxPush(3, 9, 9, stack);
      stack.push(9);
```

Run or submit code when you're ready.