Solution 1

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

using System;
using System.Collections.Generic;

public class Program {
  public class MinMaxStack {
    List<Dictionary<string, int> > minMaxStack = new List<Dictionary<string, int>
    List<int> stack = new List<int>();

    // O(1) time | O(1) space
    public int Peek() {
      return stack[stack.Count - 1];
    }

    // O(1) time | O(1) space
    public int Pop() {
      minMaxStack.RemoveAt(minMaxStack.Count - 1);
      var val = stack[stack.Count - 1];
      stack.RemoveAt(stack.Count - 1);
      return val;
    }

    // O(1) time | O(1) space
    public void Push(int number) {
      Dictionary<string, int> newMinMax = new Dictionary<string, int>();
      newMinMax.Add("min", number);
      newMinMax.Add("max", number);
      if (minMaxStack.Count > 0) {
        Dictionary<string, int> lastMinMax = new Dictionary<string, int>(
          minMaxStack[minMaxStack.Count - 1]
          );
        newMinMax["min"] = Math.Min(lastMinMax["min"], number);
        newMinMax["max"] = Math.Max(lastMinMax["max"], number);
      }
      minMaxStack.Add(newMinMax);
      stack.Add(number);
    }

    // O(1) time | O(1) space
    public int GetMin() {
      return minMaxStack[minMaxStack.Count - 1]["min"];
    }

    // O(1) time | O(1) space
    public int GetMax() {
      return minMaxStack[minMaxStack.Count - 1]["max"];
    }
  }
}

```

Solution 1 | Solution 2 | Solution 3

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

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


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


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


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

```

Run or submit code when you're ready.