MOVIKA BHAGAVAKAR
movikabh@gmail.com

ASSIGNMENT-DAY-6

Question 1

Write a function to find the maximum element in the stack.

```
// C++ program to keep track of maximum
// element in a stack
#include <bits/stdc++.h>
using namespace std;

class StackWithMax
{
        // main stack
        stack<int> mainStack;

        // stack to keep track of max element
        stack<int> trackStack;

public:
        void push(int x)
        {
                mainStack.push(x);
                if (mainStack.size() == 1)
                {
                        trackStack.push(x);
                        return;
                }

                // If current element is greater than
                // the top element of track stack, push
                // the current element to track stack
                // otherwise push the element at top of
                // track stack again into it.
                if (x > trackStack.top())
                        trackStack.push(x);
                else
                        trackStack.push(trackStack.top());
        }

        int getMax()
```

MOVIKA BHAGAVAKAR
movikabh@gmail.com

ASSIGNMENT-DAY-6

```
        {
                return trackStack.top();
        }

        int pop()
        {
                mainStack.pop();
                trackStack.pop();
        }
};

// Driver program to test above functions
int main()
{
        StackWithMax s;
        s.push(20);
        cout << s.getMax() << endl;
        s.push(10);
        cout << s.getMax() << endl;
        s.push(50);
        cout << s.getMax() << endl;
        return 0;
}
```

Question 2

Write a function to find the minimum element in the stack.

```
// C++ program to implement a stack that supports
// getMinimum() in O(1) time and O(1) extra space.
#include <bits/stdc++.h>
using namespace std;

// A user defined stack that supports getMin() in
// addition to push() and pop()
struct MyStack
{
        stack<int> s;
        int minEle;
```

MOVIKA BHAGAVAKAR
movikabh@gmail.com

ASSIGNMENT-DAY-6

```
// Prints minimum element of MyStack
void getMin()
{
	if (s.empty())
		cout << "Stack is empty\n";

	// variable minEle stores the minimum element
	// in the stack.
	else
		cout <<"Minimum Element in the stack is: "
			<< minEle << "\n";
}

// Prints top element of MyStack
void peek()
{
	if (s.empty())
	{
		cout << "Stack is empty ";
		return;
	}

	int t = s.top(); // Top element.

	cout << "Top Most Element is: ";

	// If t < minEle means minEle stores
	// value of t.
	(t < minEle)? cout << minEle: cout << t;
}

// Remove the top element from MyStack
void pop()
{
	if (s.empty())
	{
		cout << "Stack is empty\n";
		return;
```

MOVIKA BHAGAVAKAR
movikabh@gmail.com

ASSIGNMENT-DAY-6

```
		}

		cout << "Top Most Element Removed: ";
		int t = s.top();
		s.pop();

		// Minimum will change as the minimum element
		// of the stack is being removed.
		if (t < minEle)
		{
			cout << minEle << "\n";
			minEle = 2*minEle - t;
		}

		else
			cout << t << "\n";
}

// Removes top element from MyStack
void push(int x)
{
		// Insert new number into the stack
		if (s.empty())
		{
			minEle = x;
			s.push(x);
			cout << "Number Inserted: " << x << "\n";
			return;
		}

		// If new number is less than minEle
		if (x < minEle)
		{
			s.push(2*x - minEle);
			minEle = x;
		}

		else
```

MOVIKA BHAGAVAKAR
movikabh@gmail.com

ASSIGNMENT-DAY-6

```
                s.push(x);


                cout << "Number Inserted: " << x << "\n";
        }
};


// Driver Code
int main()
{
        MyStack s;
        s.push(3);
        s.push(5);
        s.getMin();
        s.push(2);
        s.push(1);
        s.getMin();
        s.pop();
        s.getMin();
        s.pop();
        s.peek();

        return 0;
}
```