Data Structure (Lab Performance- 03)

Stacks and Applications

(Time: 2 Hours)

1. Implement a Stack Data Structure based on the following functions:

```
Int Stack[100], Top=0, MaxSize=100; // Stack holds the elements;
// Top is the index of Stack always pointing to the first/top element of the stack.
bool IsEmpty(s); // returns True if stack has no element
bool IsFull(); // returns True if stack full
bool Push( const int Element ); // inserts Element at the top of the stack
bool Pop( int &Element ); // deletes top element from stack into Element
bool TopElement( int &Element ); // gives the top element in Element
void Show(); // prints the whole stack
```

2. Create a class Stack following the given outline:

```
class MyStack{
        int Stack[100], Top, MaxSize;
        public:
        //Initializing stack
        MyStack( int Size = 100 )
        {
                MaxSize = Size; Top = 0;
        }
        bool IsEmpty();
        bool IsFull();
        bool Push( const int Element );
        bool Pop( int &Element );
        bool TopElement( int &Element );
        void Show();
        void Reset(){
                Top = 0; } //Re-start the stack
};
```

3. Using the above Class of Stack Implement two new properties to the stack such that the Stack becomes dynamic in Nature.

```
MyStack( int Size = 100 ){
        MaxSize = Size; // get Size
        Stack = new int[ MaxSize ]; // create array accordingly
        Top = 0; // start the stack
}
~MyStack(){
        delete [] Stack; // release the memory for stack
}
```