Lab 6: Generics

1.	**GenericMaxStack** **Time**: 30 minutes **Problem Description** You need to create a class named GenericMaxStack that represents a last-in-first-out (LIFO) data structure with the following properties: 1. It has push(int) and pop() operations that work the same way as a normal stack 2. In addition, it has a max() operation that returns the maximum value in the current stack. 3. You have to ensure that your code is working for Integer, Double, and String data types. **Constraints** The max() operation should operate at constant complexity, O(1). This means you cannot use a loop or recursion to find the minimum value. **Test cases** 1. Push 3, 5, 2. Assert max = 5. 2. Push 2, 1, 2, 5. Pop the last element. Assert max = 2. Pop again. Assert max = 2. 3. Push 49.75, 23.54, 100.0. Assert max 100. Pop the last element. Assert max 49.75. 4. Push “OOC is bad”, “Nothing to understand”, and “Try hard”. Assert max “Try hard”. Pop the last element. Assert max “OOC is bad”. **Hint** 1. You can use the built-in Stack class if necessary. 2. You can keep up to the max in the stack each time you insert an element in the stack.	5
2.	**MinMaxStack** Time: 40 minutes You have the MaxStack class that you already created in the previous lab along with test cases in the MaxStackTest class. If you complete the first task of this lab, you also have the generic version of those classes. Your task is to create a “MinMaxStack” class with the following properties: 1. MinMaxStack has push and pop operations just like MaxStack.	5

	2. However, it does not have the minimum operation. Instead, it has an *aggregation* operation that returns either min or max based on a condition passed through the constructor. **Constraints** The *aggregate()* operation should operate at constant complexity, *O(1)*. This means you cannot use a loop or recursion to find the minimum or maximum value. **Test cases** 1. Push 3, 2, 5. Create an instance of MinMaxStack class, MinMaxStack("min"). Assert min = 2. 2. With the same data, create another instance of MinMaxStack class, MinMaxStack("max"). Assert max = 5. **Hint** 1. You can use the built-in Stack class if necessary.	