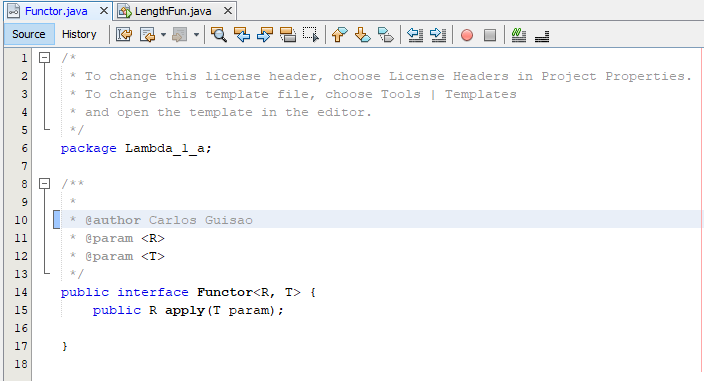
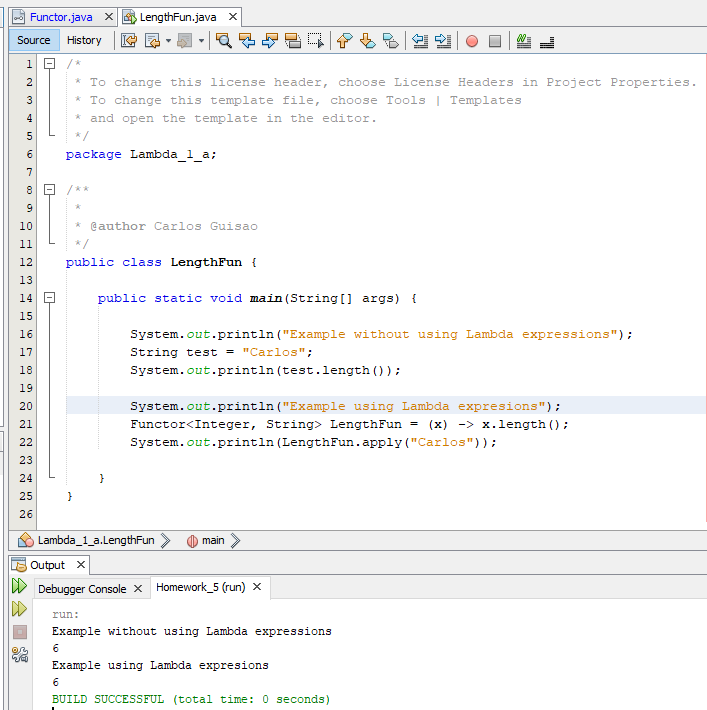
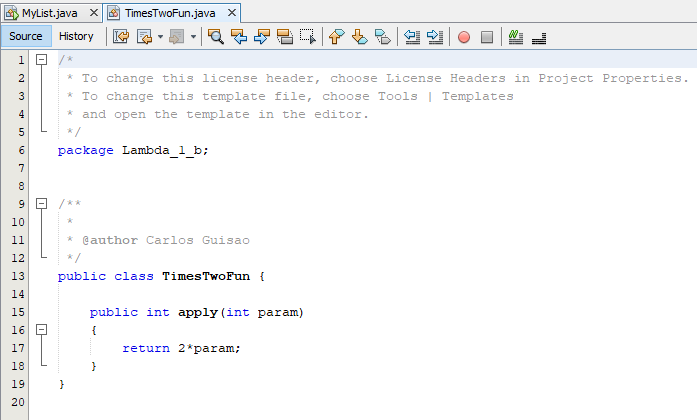
**Lambda Expressions**

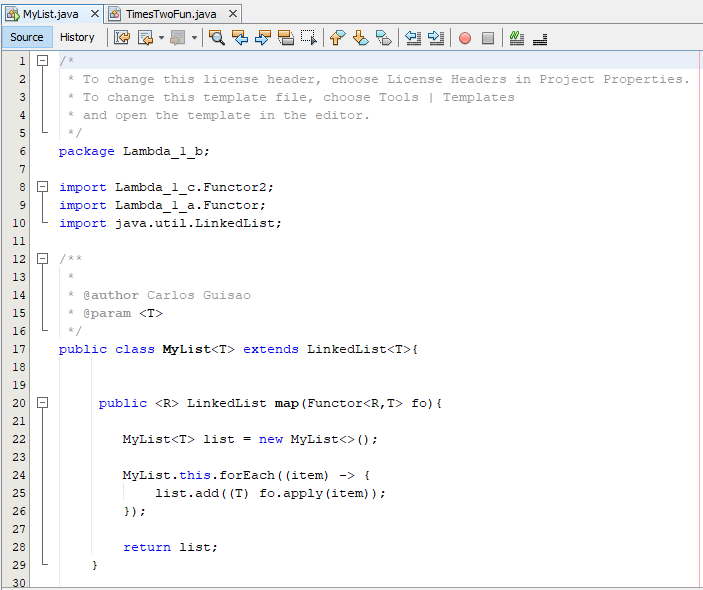
**L.1**

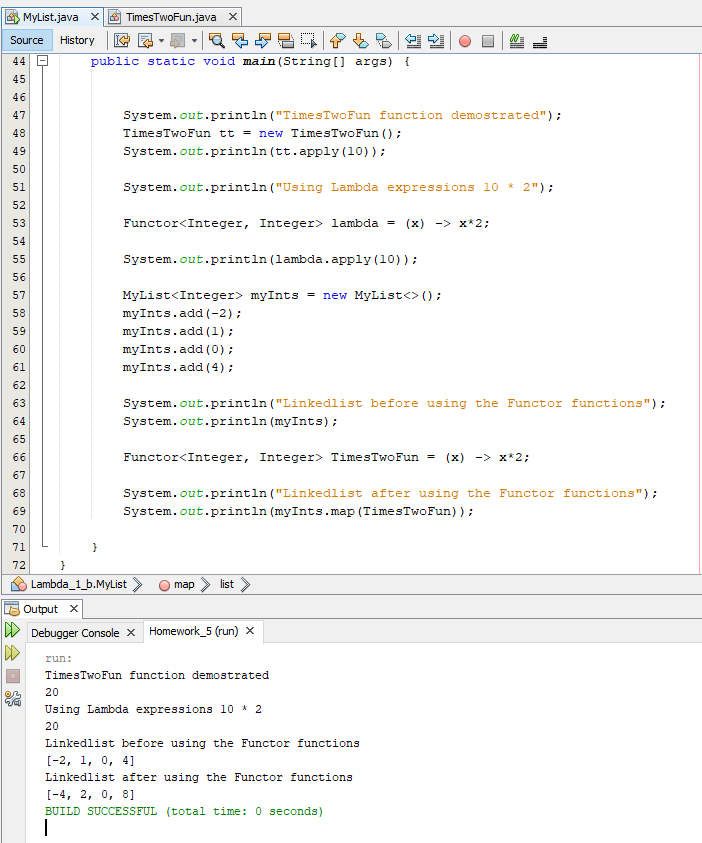




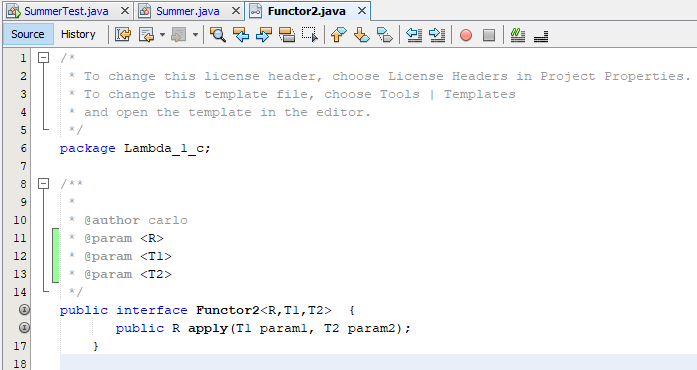
**L.1 b**

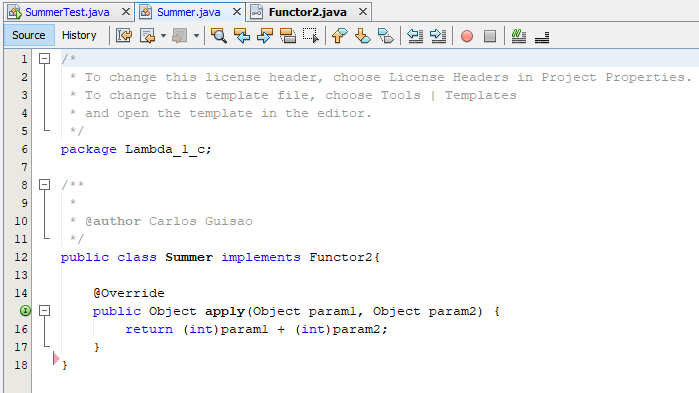


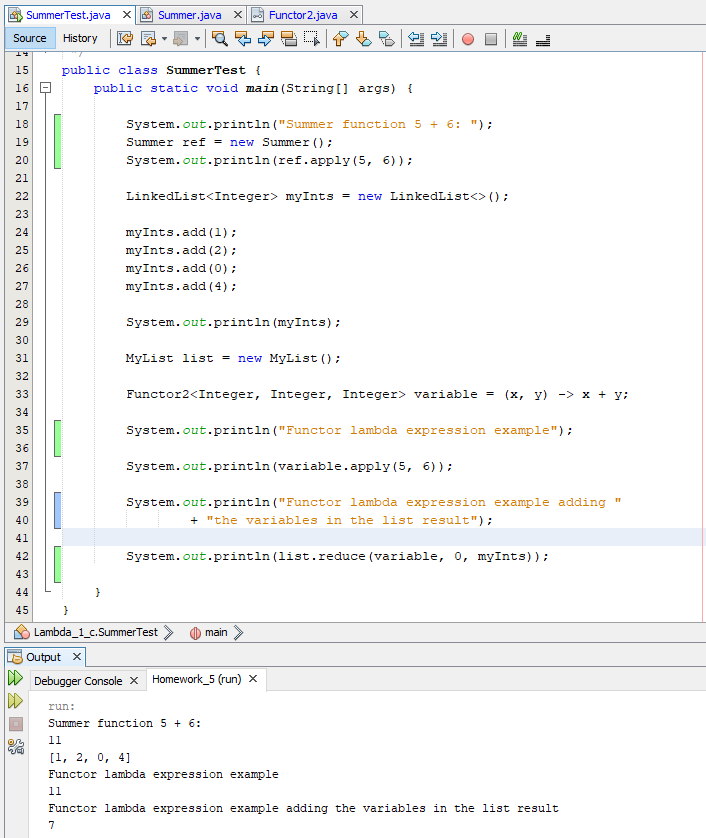




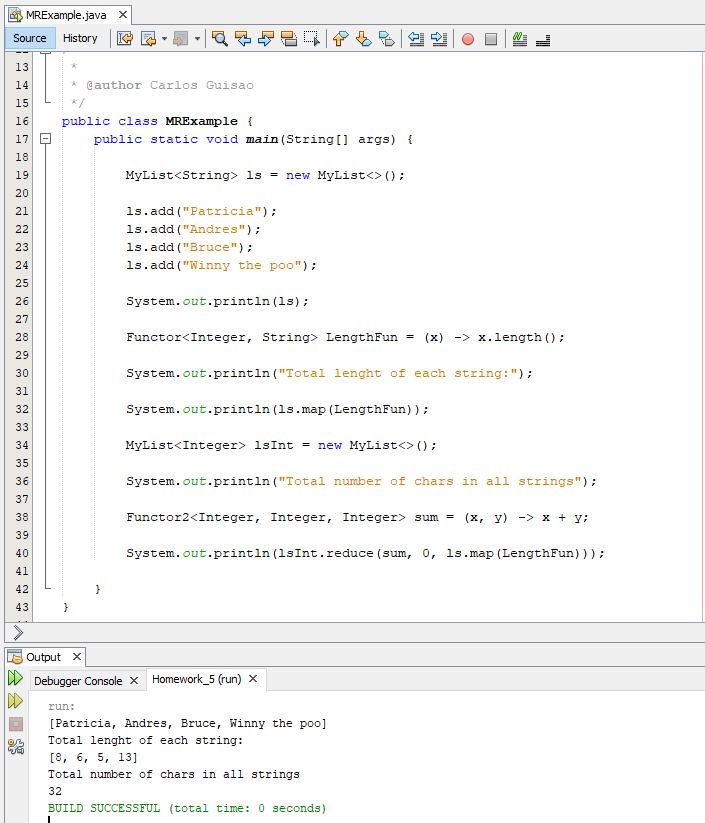
**L.1 c**



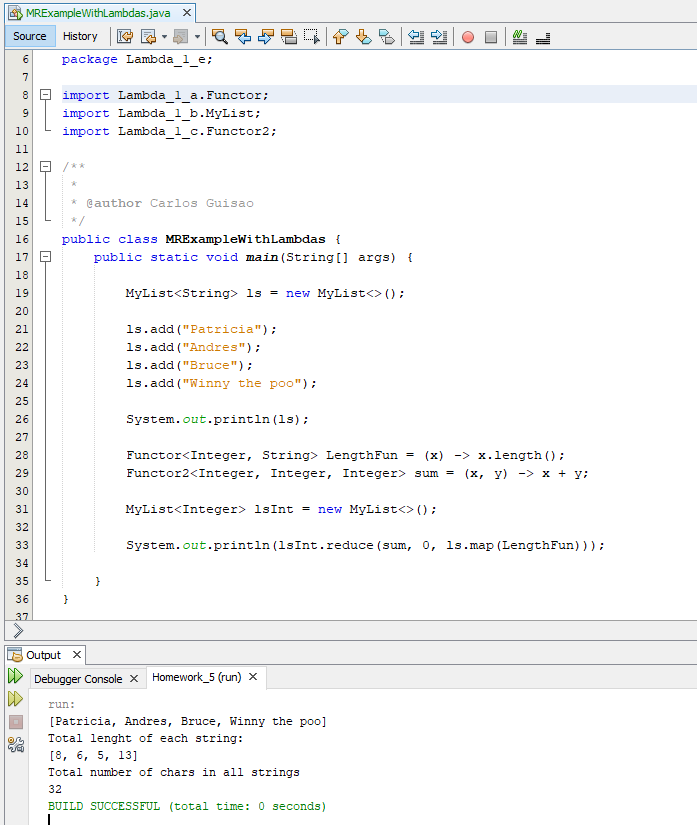




**L.1 d**

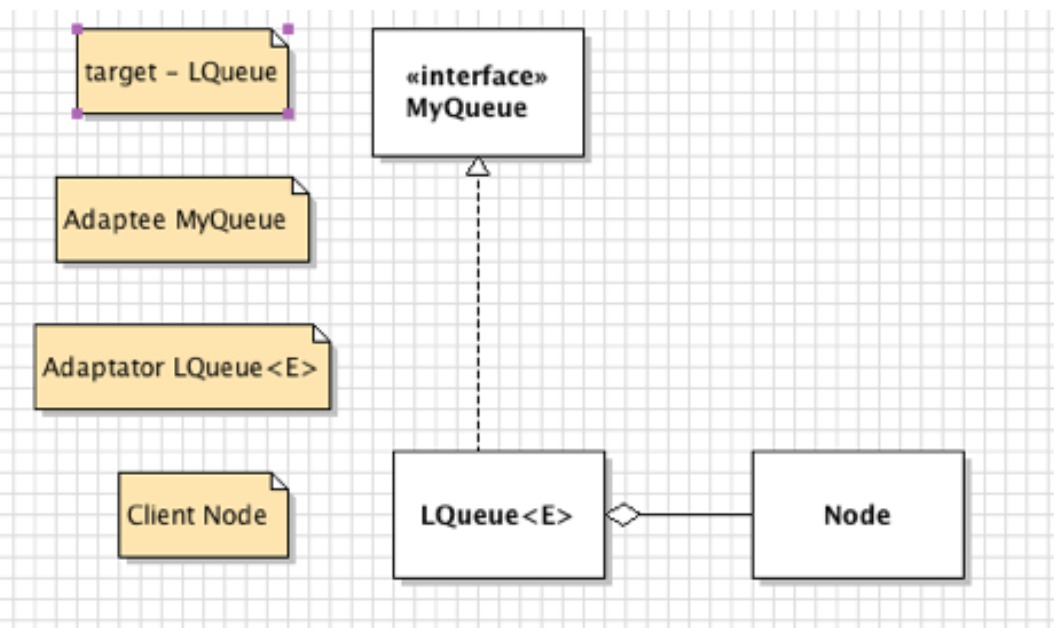


**L.1 e**

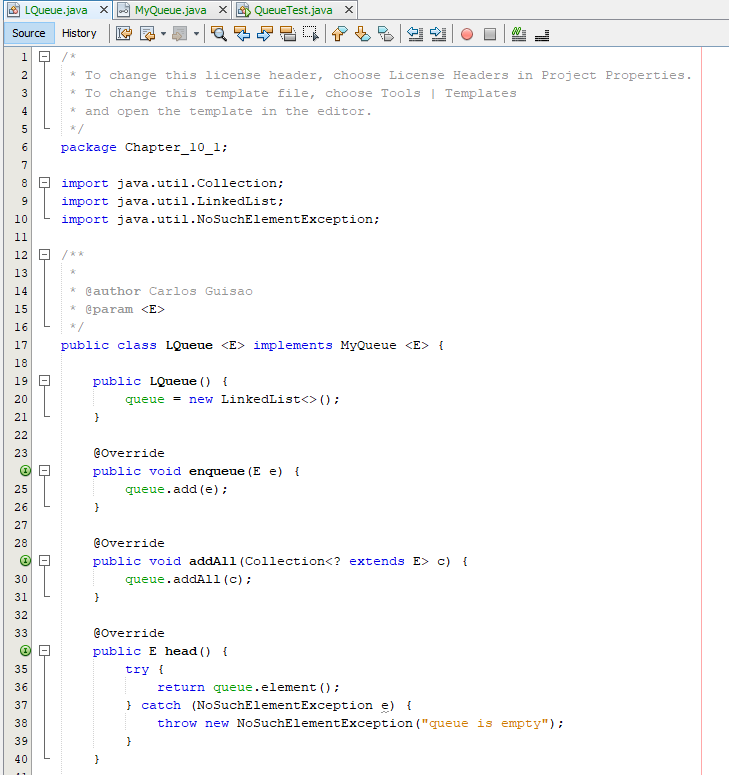


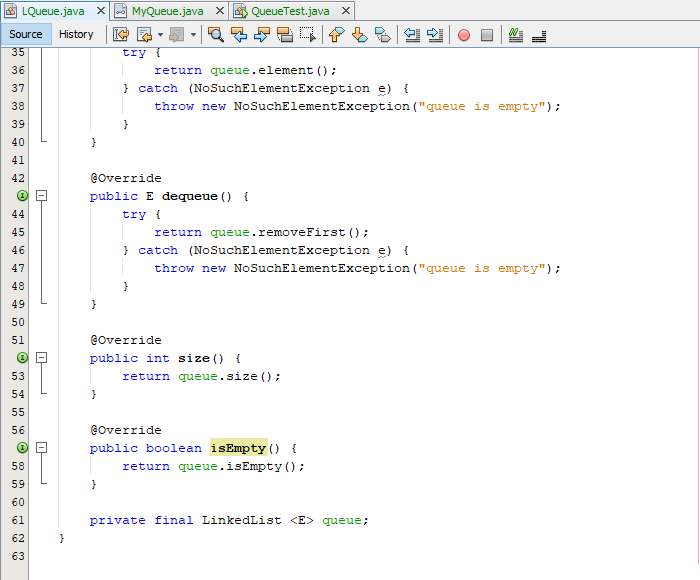
**10.1**

**a)**

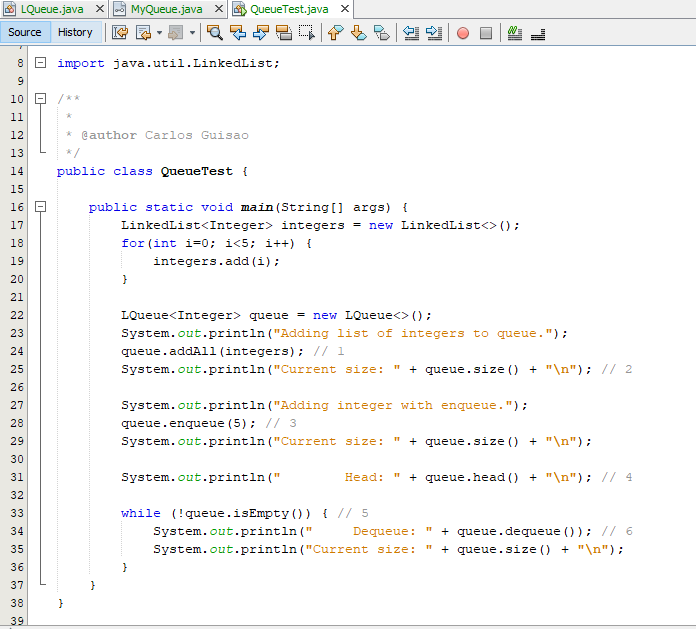


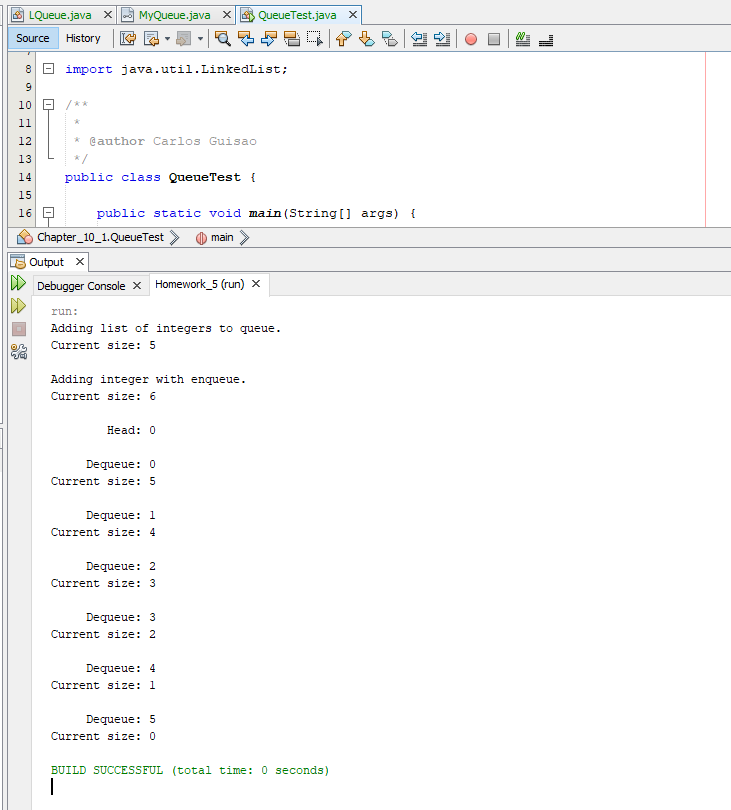
**b)**



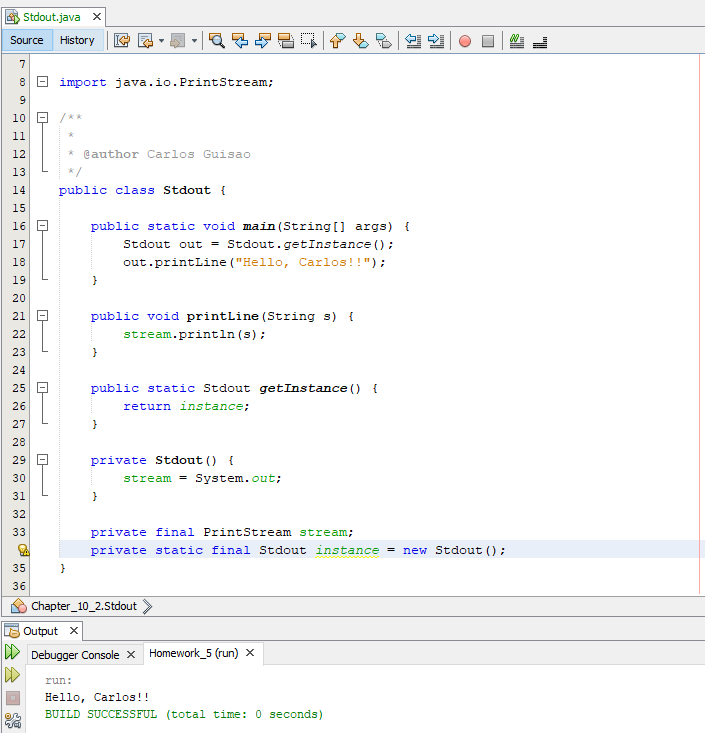


**c)**





**10.2**



**10.3**

**A -** The differences between the decorator pattern and the proxy pattern is that the decorator pattern focuses on dynamically adding functions to an object where the proxy pattern focuses on controlling access to an object or when you want to “lazyinstantiate an object.

**B-** The MouseMotionAdapter class from the Swing library is not an adapter class in the sense of the Adapter design pattern because none of its method are implemented and left empty on purpose. This class is provided as a convenience for easily creating listeners by extending this class and overriding its methods.

**7.1**

