**Exercise 3.7: Handling Void Methods with Exceptions**

**CODE:**

**Notifier Interface: -**

package org.example;  
public interface Notifier{  
 void alert(String message);  
}

**AlertService Class: -**

package org.example;  
public class AlertService {  
 private Notifier notifier;  
 public AlertService(Notifier notifier) {  
 this.notifier = notifier;  
 }  
 public void sendCriticalAlert() {  
 System.*out*.println("Attempting to send alert...");  
 notifier.alert("Critical alert!");  
 System.*out*.println("Alert sent");  
 }  
}

**AlertServiceTest Class: -**

package org.example;  
import org.junit.jupiter.api.Test;  
import static org.mockito.Mockito.\*;  
import static org.junit.jupiter.api.Assertions.\*;  
  
public class AlertServiceTest {  
 @Test  
 public void testVoidMethodThrowsException() {  
 Notifier mockNotifier = *mock*(Notifier.class);  
 *doThrow*(new RuntimeException("Notifier failed")).when(mockNotifier).alert("Critical alert!");  
  
 AlertService service = new AlertService(mockNotifier);  
 RuntimeException exception = *assertThrows*(RuntimeException.class, service::sendCriticalAlert);  
 System.*out*.println("Exception caught: " + exception.getMessage());  
  
 *verify*(mockNotifier).alert("Critical alert!");  
 System.*out*.println("Verified that alert method was called with exception.");  
 }  
}

**OUTPUT:**

