MOCKITO EXCERCISES

Exercise 1: Mocking and Stubbing

ExternalApi.java

```
public interface ExternalApi {
   String getData();
}
```

MyService.java

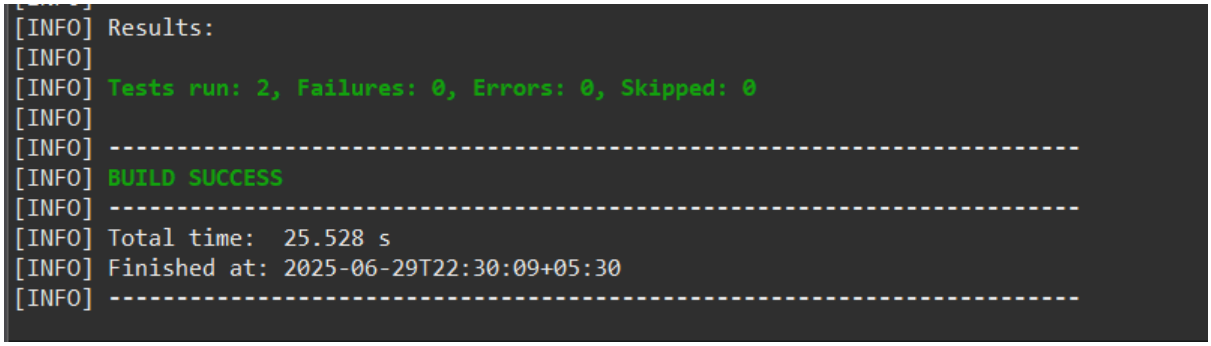
```
public class MyService {
   private ExternalApi externalApi;
   public MyService(ExternalApi externalApi) {
      this.externalApi = externalApi;
   }
   public String fetchData() {
      return externalApi.getData();
   }
}
```

MyServiceTest.java

```
import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.mockito.Mockito.*;
import org.junit.jupiter.api.Test;
public class MyServiceTest {
   @Test
   public void testExternalApi() {
      ExternalApi mockApi = mock(ExternalApi.class);
      when(mockApi.getData()).thenReturn("Mock Data");

      MyService service = new MyService(mockApi);
      String result = service.fetchData();
```

```
    assertEquals("Mock Data", result);
  }
}
```

OUTPUT:



Exercise 2: Verifying Interactions
CODE:

Calculator.java

```
package com.Math;

public interface Calculator {
        int add(int i, int j);
}
```

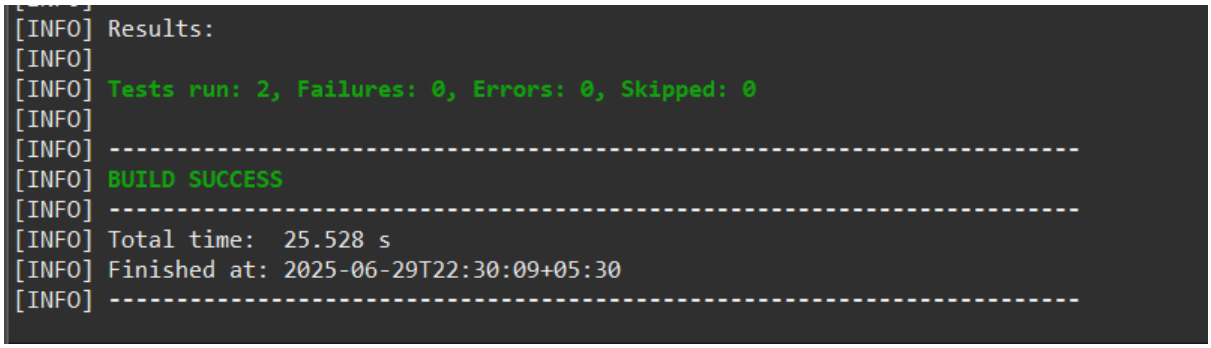
MathService.java

```
package com.Math;

public class MathService {
  private Calculator calculator;
        public MathService(Calculator calculator) {
                this.calculator=calculator;
                // TODO Auto-generated constructor stub
        }
        public int performAddition(int i, int j):
                return calculator.add(i,j);
        }
}
```

MathServiceTest.java

```
package com.Math;

import static org.mockito.Mockito.*;

import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.Test;

import org.mockito.ArgumentMatchers;

public class MathServiceTest {

  @Test

  public void testArgumentMatching() {

    Calculator mockCalculator = mock(Calculator.class);

    when(mockCalculator.add(ArgumentMatchers.eq(5),
ArgumentMatchers.eq(3))).thenReturn(8);

    MathService service = new MathService(mockCalculator);

    int result = service.performAddition(5, 3);

    assertEquals(8, result);

    verify(mockCalculator).add(5, 3);

  }

}
```

OUTPUT:



Exercise 1: Logging Error Messages and Warning Levels

CODE:

LoggingExample.java

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class LoggingExample {
   private static final Logger logger = LoggerFactory.getLogger(LoggingExample.class);

   public static void main(String[] args) {
      logger.error("This is an error message");
      logger.warn("This is a warning message");
   }
}
```

OUTPUT:

```
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4-api</artifactId>
  <version>1.7.30</version>
</dependency>
<dependency>
  <groupId>ch.qos.logback</groupId>
  <artifactId>logback-classic</artIId>
  <version>1.2.3</version>
</dependency>

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class LoggingExample {
    private static final Logger lgger = LoggerFactory

    public static void main(String[] args)
        logger.ereor("This is an error message)
    }   logger.warn("This is a warning message)
}
20:15.39.193 ERROR  LoggingExample
20:15.39.WAR WARN   LoggingExample
```