TESTING

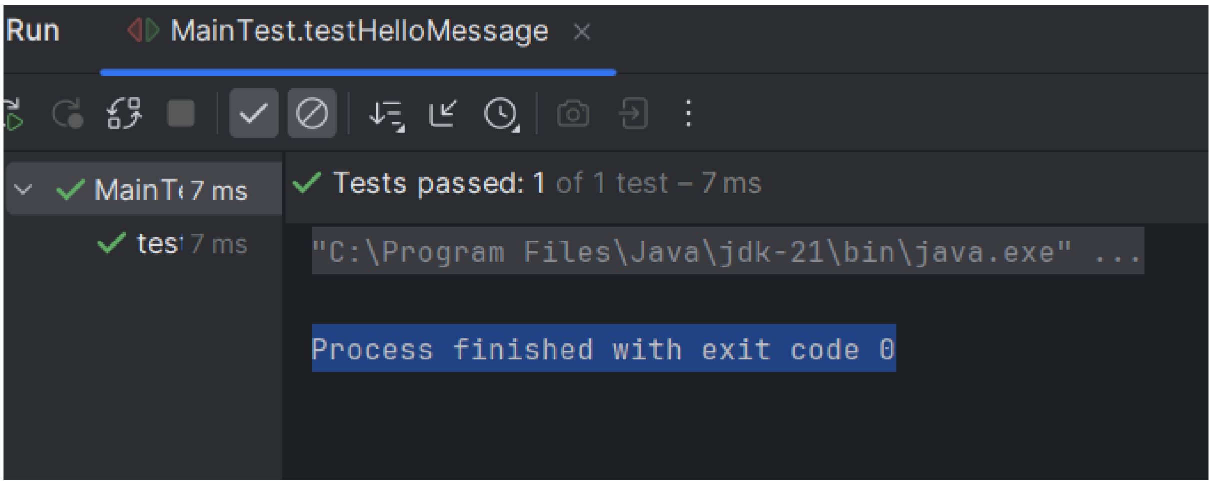
JUnit Testing Exercises

Exercise 1: Setting Up JUnit

```
<dependency>
   <groupId>junit</groupId>
   <artifactId>junit</artifactId>
   <version>4.13.2</version>
   <scope>test</scope>
</dependency>
```

```
public class MainTest {
   @Test
   public void testHelloMessage() {
       String expected = "Hello and welcome!";
       String actual = "Hello and welcome!";
       assertEquals(expected, actual);
//          assertEquals("Junit is working fine", "Junit is working fine");
   }
}
```

Output:

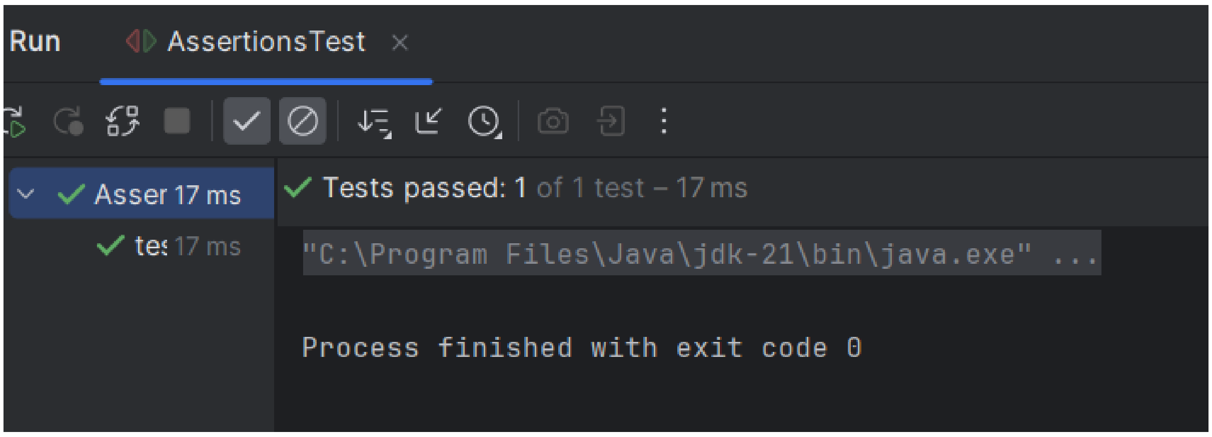


```
Tests passed: 1 of 1 test – 7 ms
"C:\Program Files\Java\jdk-21\bin\java.exe" ...

Process finished with exit code 0
```

Exercise 3: Assertions in JUnit

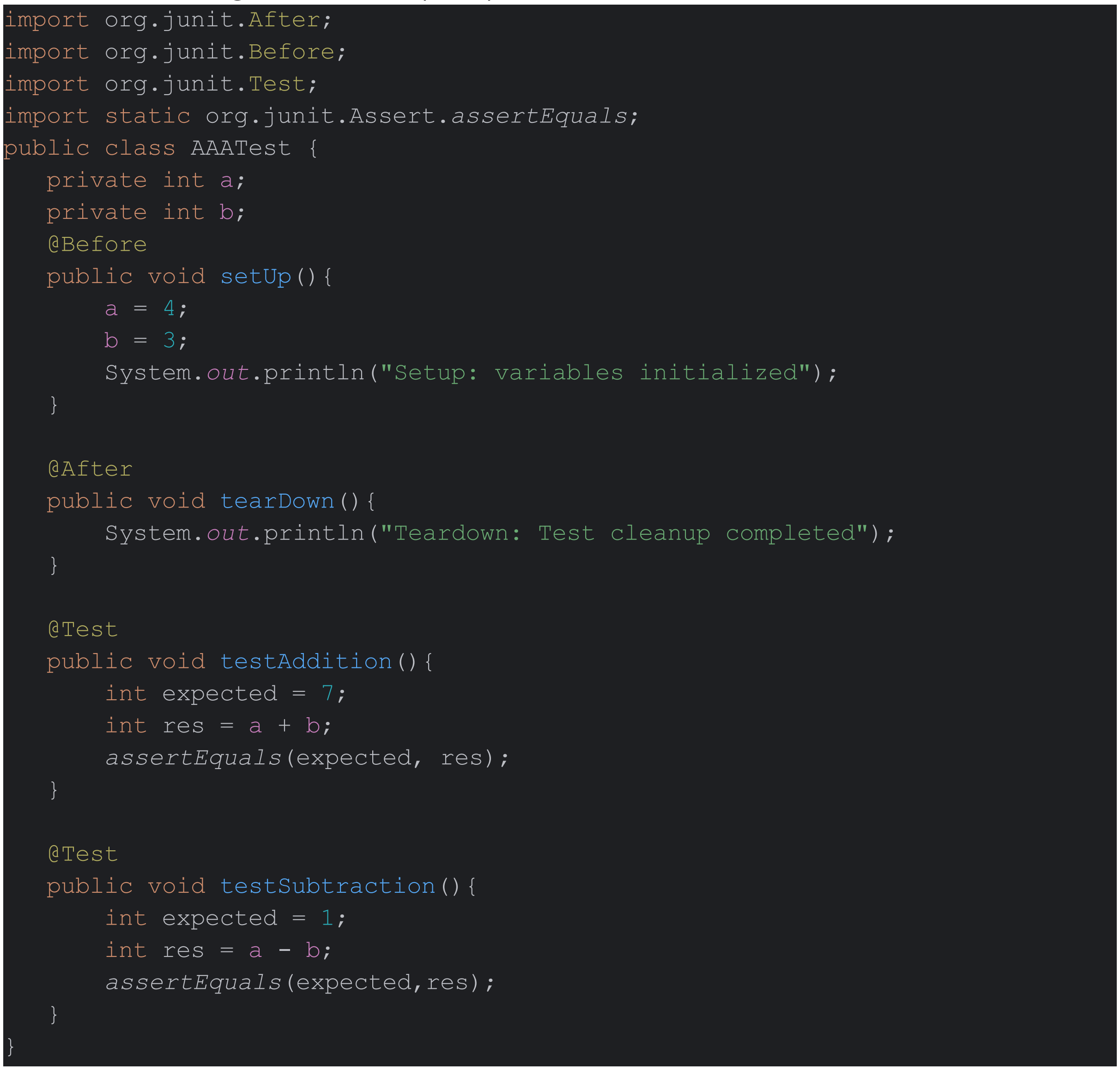
```
public class AssertionsTest {
   @Test
   public void testAssertions(){
       assertEquals(5,2+3);
       assertTrue(5>3);
       assertFalse(5 < 3);
       assertNull(null);
       assertNotNull(new Object());
   }
}
```

Output:



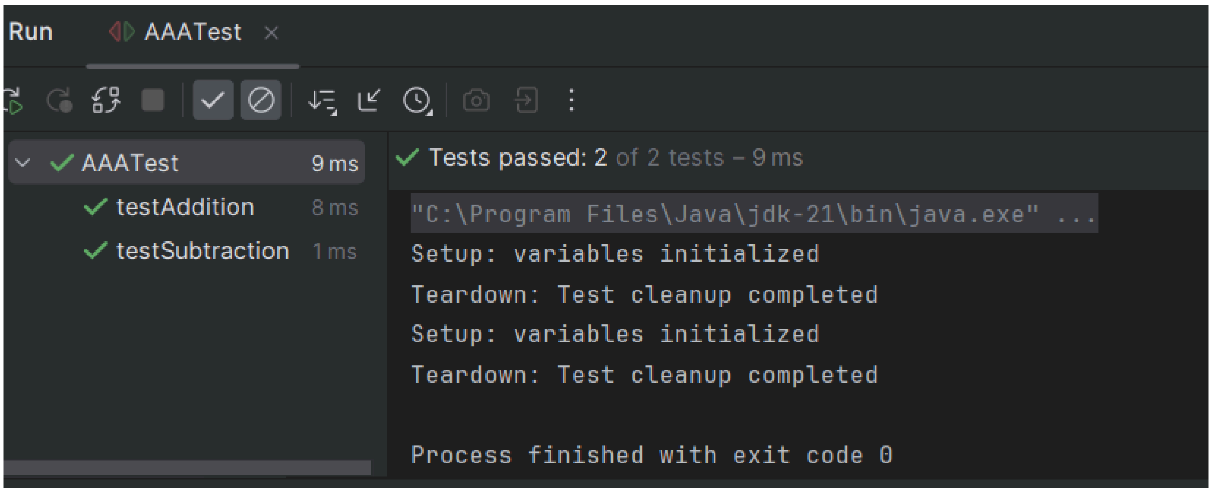
```
✓ Tests passed: 1 of 1 test – 17 ms
"C:\Program Files\Java\jdk-21\bin\java.exe" ...

Process finished with exit code 0
```

Exercise 4: Arrange-Act-Assert (AAA) Pattern



```java
import org.junit.After;
import org.junit.Before;
import org.junit.Test;
import static org.junit.Assert.assertEquals;
public class AAATest {
   private int a;
   private int b;
   @Before
   public void setUp(){
       a = 4;
       b = 3;
       System.out.println("Setup: variables initialized");
   }

   @After
   public void tearDown(){
       System.out.println("Teardown: Test cleanup completed");
   }

   @Test
   public void testAddition(){
       int expected = 7;
       int res = a + b;
       assertEquals(expected, res);
   }

   @Test
   public void testSubtraction(){
       int expected = 1;
       int res = a - b;
       assertEquals(expected,res);
   }
}
```

Output:



```
AAATest            9 ms
  testAddition     8 ms
  testSubtraction  1 ms
```

```
Tests passed: 2 of 2 tests – 9 ms
"C:\Program Files\Java\jdk-21\bin\java.exe" ...
Setup: variables initialized
Teardown: Test cleanup completed
Setup: variables initialized
Teardown: Test cleanup completed

Process finished with exit code 0
```

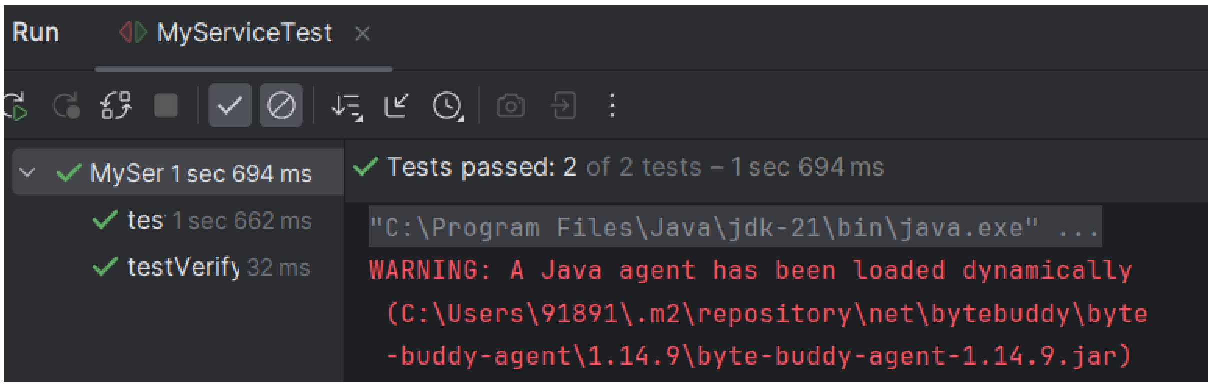
Mockito Exercises

```
<dependency>
   <groupId>org.mockito</groupId>
   <artifactId>mockito-core</artifactId>
   <version>5.7.0</version>
   <scope>test</scope>
</dependency>
```

Exercise 1&2: Mocking and Stubbing & Verifying Interactions

```
package org.example;

import static org.mockito.Mockito.*;
import org.junit.jupiter.api.Test;
import org.mockito.Mockito;
import static org.junit.Assert.*;

public class MyServiceTest {
   @Test
   public void testExternalApi() {
       ExternalApi mockApi = Mockito.mock(ExternalApi.class);
       when(mockApi.getData()).thenReturn("Mock Data");
       MyService service = new MyService(mockApi);
       String result = service.fetchData();
       assertEquals("Mock Data", result);
   }
   @Test
   public void testVerifyInteraction() {

       ExternalApi mockApi = Mockito.mock(ExternalApi.class);
       MyService service = new MyService(mockApi);
       service.fetchData();
       verify(mockApi).getData();
   }
}
```

Output:



```
MySer 1 sec 694 ms
  tes 1 sec 662 ms
  testVerify 32 ms
```

```
Tests passed: 2 of 2 tests – 1 sec 694 ms
"C:\Program Files\Java\jdk-21\bin\java.exe" ...
WARNING: A Java agent has been loaded dynamically
 (C:\Users\91891\.m2\repository\net\bytebuddy\byte
 -buddy-agent\1.14.9\byte-buddy-agent-1.14.9.jar)
```

Logging using SLF4J

```
<!-- SLF4J API -->
<dependency>
   <groupId>org.slf4j</groupId>
   <artifactId>jul-to-slf4j</artifactId>
   <version>2.0.9</version>
</dependency>

<!-- Logback (SLF4J Backend) -->
<dependency>
   <groupId>ch.qos.logback</groupId>
   <artifactId>logback-classic</artifactId>
   <version>1.4.11</version>
</dependency>
```

logback.xml

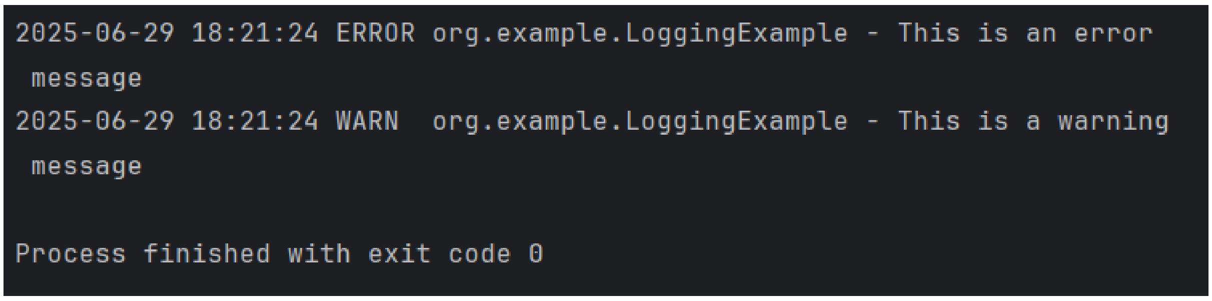
```
<configuration>
   <!-- Console appender for logging -->
   <appender name="CONSOLE" class="ch.qos.logback.core.ConsoleAppender">
       <encoder>
           <pattern>%d{yyyy-MM-dd HH:mm:ss} %-5level %logger{36} -
%msg%n</pattern>
       </encoder>
   </appender>

   <!-- Root logger configuration -->
   <root level="warn">
       <appender-ref ref="CONSOLE" />
   </root>
</configuration>
```

Exercise 1: Logging Error Messages and Warning Levels

```
package org.example;

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:

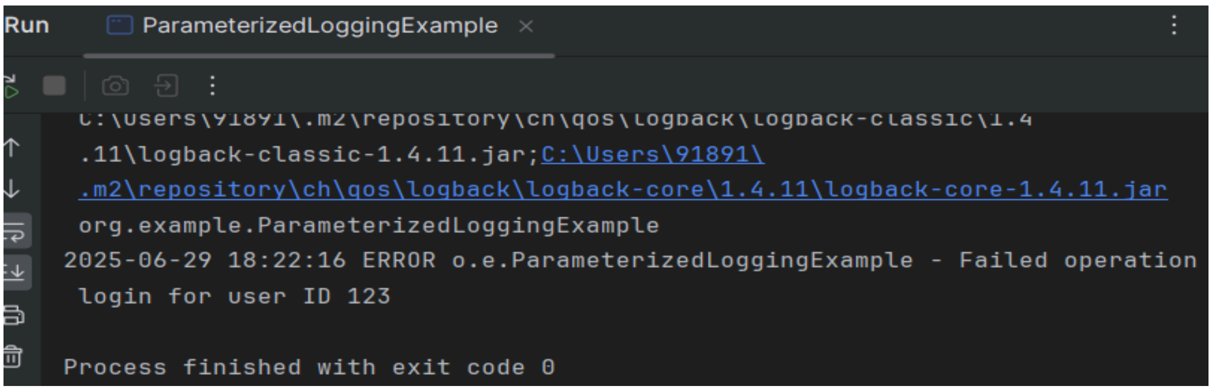


```
2025-06-29 18:21:24 ERROR org.example.LoggingExample - This is an error
 message
2025-06-29 18:21:24 WARN  org.example.LoggingExample - This is a warning
 message

Process finished with exit code 0
```

Exercise 2: Parameterized Logging

```
package org.example;

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

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

   public static void main(String[] args) {
       int userId = 123;
       String operation = "login";

       logger.info("User with ID {} performed a {} operation", userId,
operation);
       logger.error("Failed operation {} for user ID {}", operation,
userId);
   }
}
```

Output:



```
     C:\Users\91891\.m2\repository\ch\qos\logback\logback-classic\1.4
     .11\logback-classic-1.4.11.jar;C:\Users\91891\
     .m2\repository\ch\qos\logback\logback-core\1.4.11\logback-core-1.4.11.jar
     org.example.ParameterizedLoggingExample
    2025-06-29 18:22:16 ERROR o.e.ParameterizedLoggingExample - Failed operation
     login for user ID 123

    Process finished with exit code 0
```