JUnit + Spring Testing Exercises - Full Solution

# Exercise 1: Basic Unit Test for a Service Method

Service:

@Service  
public class CalculatorService {  
 public int add(int a, int b) {  
 return a + b;  
 }  
}

Test:

@SpringBootTest  
public class CalculatorServiceTest {  
  
 @Autowired  
 private CalculatorService calculatorService;  
  
 @Test  
 public void testAdd() {  
 assertEquals(5, calculatorService.add(2, 3));  
 }  
}

Output:  
Test passed, service method returned correct sum.

# Exercise 2: Mocking a Repository in a Service Test

Test:

@ExtendWith(MockitoExtension.class)  
public class UserServiceTest {  
  
 @Mock  
 private UserRepository userRepository;  
  
 @InjectMocks  
 private UserService userService;  
  
 @Test  
 public void testGetUserById() {  
 User user = new User();  
 user.setId(1L);  
 user.setName("John");  
  
 when(userRepository.findById(1L)).thenReturn(Optional.of(user));  
  
 User result = userService.getUserById(1L);  
 assertEquals("John", result.getName());  
 }  
}

Output:  
Mocked repository returned expected user.

# Exercise 3: Testing a REST Controller with MockMvc

@WebMvcTest(UserController.class)  
public class UserControllerTest {  
  
 @Autowired  
 private MockMvc mockMvc;  
  
 @MockBean  
 private UserService userService;  
  
 @Test  
 public void testGetUser() throws Exception {  
 User user = new User();  
 user.setId(1L);  
 user.setName("Alice");  
  
 when(userService.getUserById(1L)).thenReturn(user);  
  
 mockMvc.perform(get("/users/1"))  
 .andExpect(status().isOk())  
 .andExpect(jsonPath("$.name").value("Alice"));  
 }  
}

Output:  
GET endpoint returned user as JSON successfully.

# Exercise 4: Integration Test with Spring Boot

@SpringBootTest  
@AutoConfigureMockMvc  
public class IntegrationTest {  
  
 @Autowired  
 private MockMvc mockMvc;  
  
 @Autowired  
 private UserRepository userRepository;  
  
 @BeforeEach  
 public void setup() {  
 User user = new User();  
 user.setId(1L);  
 user.setName("TestUser");  
 userRepository.save(user);  
 }  
  
 @Test  
 public void testGetUserIntegration() throws Exception {  
 mockMvc.perform(get("/users/1"))  
 .andExpect(status().isOk())  
 .andExpect(jsonPath("$.name").value("TestUser"));  
 }  
}

Output:  
Integration test passed; flow from controller to DB verified.