public class LoginTest {

@Test

public void testCheckUserNameReturnsTrueWhenUsernameIsCorrectlyFormatted() {

Login login = new Login("kyl\_1", "Ch&&sec@ke99!", "Kyle", "Smith");

assertTrue(login.checkUserName());

}

@Test

public void testCheckUserNameReturnsFalseWhenUsernameIsIncorrectlyFormatted() {

Login login = new Login("kyle!!!!!!!", "Ch&&sec@ke99!", "Kyle", "Smith");

assertFalse(login.checkUserName());

}

@Test

public void testCheckPasswordComplexityReturnsTrueWhenPasswordIsComplex() {

Login login = new Login("kyl\_1", "Ch&&sec@ke99!", "Kyle", "Smith");

assertTrue(login.checkPasswordComplexity());

}

@Test

public void testCheckPasswordComplexityReturnsFalseWhenPasswordIsNotComplex() {

Login login = new Login("kyl\_1", "password", "Kyle", "Smith");

assertFalse(login.checkPasswordComplexity());

}

@Test

public void testLoginUserReturnsTrueWhenCredentialsAreCorrect() {

Login login = new Login("kyl\_1", "Ch&&sec@ke99!", "Kyle", "Smith");

assertTrue(login.loginUser("kyl\_1", "Ch&&sec@ke99!"));

}

@Test

public void testLoginUserReturnsFalseWhenCredentialsAreIncorrect() {

Login login = new Login("kyl\_1", "Ch&&sec@ke99!", "Kyle", "Smith");

assertFalse(login.loginUser("kyl\_1", "password"));

}

@Test

public void testReturnLoginStatusReturnsSuccessfulLoginMessageWhenCredentialsAreCorrect() {

Login login = new Login("kyl\_1", "Ch&&sec@ke99!", "Kyle", "Smith");

assertEquals("