**Mandatory tasks**

**Task 1.1** Write check list for login page testing for <https://www.wiggle.com/login>.

The check list should contain 5-6 ideas for future test-cases NOT included in the smoke test.

**Task 1.2.** - create locators for UI elements to cover test cases from check list for login page testing for <https://www.wiggle.com/login>. (e.g. from check list of task 1.2)

**Task 1.3** - write test cases using Gherkin language (e.g. https://cucumber.io/docs/gherkin/reference/) for points from the check list in Task1.1.

**Task 2 - unit tests**.

Given: two circles centers of which belong to axis X. (e.g. (x1; 0; r1); (x2; 0; r2))

The user inputs positive integer numbers x1; r1; x2; r2 from the keyboard.

It is necessary to determine the quantity of common points of the circles.

Please, output the result as a text in the console

1. Perform the domain testing based on the meaning of the task.

Your decision form in the table. Select and write the values ​​that you will use for your future tests.

1. Implement your java source to resolve the task. Numbers should be entered via console

Implement unit-tests for the all your java methods using unit tests frameworks (e.g. TestNG, Junit) and values selected through domain testing completed previously (1 (a)). Include parametrized tests into scope of your unit-tests

**Framework**

**Task 3 (Set up the framework. Create Factory package. Implement 2 test-cases in the Factory package)**

Get the source from <https://github.com/School-of-Digital-Competencies/autoframeWork-template-Java_2025>

(Create own branch or fork the source, do NOT push smth into master, please)

**Set-up**

Set up the framework template shared with you with the help of Training Autoframework tutorial.docx (see telegram chart or Moodle , lesson 6). Try to build (re-build) the project and run test method from MainTest class or run MainTest class in the whole

**Framework**

**Factory**

Create a new package: src/test/java/com/stv/factory.

Add into it two inner packages: src/test/java/com/stv/factory/factorypages and

src/test/java/com/stv/factorytests.

Implement appropriate classes according to **Page Factory** pattern. Using such pattern, implement any two test-cases from your test suit (Task 1) for <https://www.wiggle.com/login>

**Please consider the following general points:**

* use @FindBy and @FindBys
* internal page content is hidden (private)
* public methods implements services (which a particular page offers)
* page’s methods return new page objects
* all verification (assertions) are made out of the page

**Task 4**

**Framework – BDD – Common task**

*Given:* Framework has been set up, factory package has been added and appropriated factory PO classes have been implemented

*Required:* 1) Create BDD package

2) Implement 2 cucumber scenarios. The first scenario (simple) should be from your test suit (please see Task1\_3). The second should be a scenario outline, you can cover any feature you wish

*Target website for testing:* https://www.wiggle.com/

*Useful info* can be found from Lesson\_9\_Introduction\_into\_BDD.pptx (you can download it from Moodle, topic 9 )

***Note:***

Please, do NOT send your answers. The task will be verified at the lessons only

**Task 5**

**Framework – Test cases to describe a Defect**

*Target website for testing:* <https://www.wiggle.com/>

Required: find a defect (or create an enhancement instead) and implement appropriate auto test

You can use any template from your study project (design, factory or bdd)

***Note:***

Please, do NOT send your answers. The task will be verified at the lessons only

**Task 6. Framework – BDD – Individual task**

*Given:* Framework has been set up, factory package has been added, and appropriate factory PO classes have been implemented, BDD package has been created

*Required:*

Implement a cucumber scenario in compliance with the user case given by the teacher.

*Target website for testing:* <https://www.wiggle.com/>

*Useful info* can be found from Lesson\_9\_Introduction\_into\_BDD.pptx (you can download it from Moodle, topic 9 or a common telegram chat

*Deadline:* you can spend the time for the task until the exam lesson (10th of June, 2025)

***Note*:** if you missed the previous lesson and haven’t received the individual task yet, you can drop me a direct message in telegram and we will make arrangements.

Please, do NOT send your answers. The task will be verified at the lessons only

**Task 7**

**API testing task**

With the help of “Lesson\_10\_Introduction into Automated testing of Web Services with Java .pptx” select any GET request <https://jsonplaceholder.typicode.com/> implement 2 positive test-cases and 2 negative test-cases. Besides the status validation, please check something else from the response (according to your wishes and possibilities)

You can create a separate maven project and use

*<!-- 1. Rest assured library -->  
 <!-- https://mvnrepository.com/artifact/io.rest-assured/rest-assured -->* <dependency>  
 <groupId>io.rest-assured</groupId>  
 <artifactId>rest-assured</artifactId>  
 <version>5.0.0</version>  
 <scope>test</scope>  
 </dependency>

**Optional tasks:**

To increase your final score, you can add logs, implement runners, add reporting into your project