Telit Cinterion

Java and Selenium Assignment

Part 1

Java Homework Assignment

Objective:
The goal of this assignment is to reinforce your understanding of core Java concepts, object-oriented programming (OOP), and problem-solving skills. You will develop a simple application, implement classes, and practice using collections and exception handling.

Part 1: Basic Java Concepts

1. Create a Java Console Application:
- Start by creating a new Java project named `StudentManagementSystem`.
- Inside this project, create a main class called `Main`.

2. Implement a `Student` Class:
- Define a `Student` class with the following attributes:
  - `String name`
  - `int age`
  - `String studentId`
  - `List<String> courses`

- Implement the following methods:
  - A constructor to initialize `name`, `age`, and `studentId`.
  - A method `void enrollCourse(String course)` to add a course to the `courses` list.
  - A method `void printStudentInfo()` to display the student’s name, age, student ID, and a list of enrolled courses.

3. Object Creation and Management:
- In the `Main` class, instantiate three `Student` objects, each with different details.
- Enroll each student in 2–3 different courses.
- Print the details of each student using the `printStudentInfo()` method.

Part 2: Collections and Exception Handling

4. Manage a List of Students:
- In the `Main` class, create a `List<Student>` to store the student objects.
- Add the three `Student` objects to this list.
- Write a method `Student findStudentById(String studentId)` that searches for a student in the list by their `studentId` and returns the `Student` object. If the student is not found, throw an appropriate exception with a custom message.

5. Exception Handling:
- Modify the `Main` class to handle the exception thrown by `findStudentById`.
- If an exception occurs, display a message like: "Student with ID [ID] not found."

Part 3: Optional (Bonus)

6. Advanced Features:
- Implement sorting of students by their `name` using `Collections.sort()` and a custom `Comparator`.
- Allow users to input new students and enroll in courses through the console.

Submission Guidelines:

- Submit your Java project as a ZIP file, including all `.java` files and a README explaining how to run the program.
- Ensure your code is well-commented and follows Java naming conventions.

Part 2

Selenium Homework Assignment

Objective:
This assignment evaluates your advanced knowledge of Selenium WebDriver, automation frameworks, and test automation best practices. You will create an automated test suite, implement a custom reporting mechanism, and optimize your test execution strategies.

Part 1: Setting Up the Automation Framework

1. Framework Setup:
- Create a Selenium-based test automation framework using your preferred programming language (e.g., Java).
- Use Maven or Gradle for managing Java dependencies.
- Implement the Page Object Model (POM) design pattern to structure and organize your test code.

Part 2: Writing Test Cases

2. Automate the Following Test Scenarios on an E-Commerce Website:
- Use a demo e-commerce site such as Automation Practice (http://automationpractice.pl) or a similar site.

  1. User Login: Test the login functionality with both valid and invalid credentials.
  2. Product Search: Automate the process of searching for a product, verify that results are displayed correctly, and ensure the correct product can be selected.
  3. Add to Cart: Automate adding a product to the cart and verify that the cart updates accordingly.
  4. Checkout Process: Automate the checkout process, including entering shipping information, selecting payment methods, and confirming the order.

- Ensure tests are data-driven by using an external data source (e.g., Excel, JSON, or CSV files).

Part 3: Cross-Browser Testing

3. Cross-Browser Testing:
- Implement cross-browser testing to run your tests on at least three different browsers (e.g., Chrome, Firefox, Edge).
- Set up your framework to support running tests in parallel across different browser instances.

Submission Guidelines:

- Submit your automation framework as a ZIP file with all relevant code, configuration files, and a README explaining how to run the tests.
- Include reports generated from the test execution.
- Ensure your code follows industry best practices, is well-documented, and adheres to naming conventions.