1. Implicit vs Explicit Wait

Implicit wait – tells WebDriver to poll the DOM for a certain amount of time when trying to find *any* element.  
Explicit wait – waits for a specific *condition* (e.g. element clickable, visible) before proceeding.  
Use implicit for general waits, explicit for specific elements.

2. findElement vs findElements

* findElement – throws NoSuchElementException if nothing found.
* findElements – returns an empty list if no elements found (no exception).

3. NoSuchElementException vs TimeoutException

* NoSuchElementException – element not found *immediately* (no wait applied).
* TimeoutException – explicit wait *expired* before condition became true.

4. StaleElementReferenceException

Causes:

* DOM updated or page reloaded.
* Element removed or re-rendered.
* Switched to another frame or window.

Fixes:

* Re-locate the element after reload.
* Use WebDriverWait until the element is stable.
* Catch the exception and retry.
* Apply Page Object Model to manage element lookups dynamically.

5. CSS vs XPath

* CSS is faster, simpler, and natively supported by browsers.
* XPath is more flexible but slower and breaks easily if page structure changes.  
  ➡️ CSS is preferred ~80% of the time for stability and speed.

6. Page Object Model (POM)

Design pattern that separates page structure (locators) and tests (logic).  
Each page is represented by a class with web elements and actions.  
Benefits: code reuse, easy maintenance, cleaner tests.

7. Headless Chrome – Common Issues

* Missing visual elements or animations not rendered.
* Some JS or pop-ups behave differently.
* Screenshots may look blank or incomplete.
* Requires manual debugging with normal (headed) mode first.

8. Implicit + Explicit Wait Together

They add up, causing unexpected long delays.  
Explicit waits internally call findElement, which is affected by implicit wait → double waiting.  
Use only one type of wait per test.

9. Keep Chrome open after test

Use ChromeOption:

ChromeOptions options = new ChromeOptions();

options.setExperimentalOption("detach", true);

This prevents the browser from closing when the test ends.

10. Java: == vs .equals() (Strings)

* == → compares references (memory addresses).
* .equals() → compares actual text content.  
  → true for same reference with ==, or identical text with .equals().

11. List vs ArrayList

Use List type for flexibility (interface).  
ArrayList is one concrete implementation.  
Declaring variables/arguments as List allows switching to another list type later.

12. JUnit: assertEquals vs assertTrue

* assertEquals(expected, actual) → compares specific values.
* assertTrue(condition) → checks a boolean expression.  
  Use assertEquals when comparing results; assertTrue for logical conditions.

13. visibilityOfElementLocated vs presenceOfElementLocated

* presenceOfElementLocated → element exists in the DOM (can be hidden).
* visibilityOfElementLocated → element is *visible* (displayed and not hidden).

14. Maven Surefire – run single test

Run one class:

mvn -Dtest=TestClassName test

Run one method:

mvn -Dtest=TestClassName#methodName test

15. data-\* attributes for locators

Use data-\* attributes (like data-test="login-button") for stable and test-friendly selectors.  
They are not affected by UI or styling changes, making tests more reliable.