# TESTNG

## Advantages of TestNG or Why we go for TestNG:

- It provides default HTML report
  Once the execution is completed, the report will be generated inside the test output folder.
- We can pass the input using parameters concept
  We can pass the input using parameters tag, then in testng.XML file we have to pass the parameter tag with name & value and then we have to call the name in the main execution class.
- We can change the order of execution by providing priority to each test cases using priority concept
  The default value of the execution is 0. The execution flows in negative to positive.
- We can pass bulk of datas using data provider concept
  We can pass bulk of data using data provider where we need to provide name to each test cases and call the name in the execution using @dataprovider annotation.
- We can achieve parallel execution
  We can execute parallelly in 3 ways like class, method and test.
- We can achieve cross browser execution
  We have to pass the parameter tag in testng.xml by providing different browser value, then we need to call the name in the execution using @parameter annotation.
- We can execute particular test cases using grouping concept
  We need to name each test cases and we have to call the name in the testng.xml file using Include (or) Exclude tag name.
- We can achieve both hard assert and soft assert
  For verification and validation purpose we are going with assert concept.
- We can rerun the failed test cases
  We can rerun the failed test cases using three ways:
  - By Manual: All the failed test cases will be stored inside the test output folder. We can execute the failed test cases manually from there.
  - By Automation: Here we know which test cases are getting failed. We need to implement IRetryAnalyzer interface by creating a class and then rerun failed test cases using retry() method.
  - By Automation: Here we don't know which test cases are getting failed. We have to add listeners tag in testNG.xml file, then in listeners we have to mention class name where we need to implement transform() method from the IAnnotationTransformer interface.
    Using getRetryAnalayzer() method, we can get the failed test cases from listener and using setRetryAnalyzer() we can trigger the IRetryAnalyzer interface to rerun the failed test cases.

**Annotations used in TestNG:**

- @BeforeSuite: Will execute before all tests in the suite.
- @AfterSuite: Will execute after all tests in the suite.
- @BeforeTest: Will execute before any test method belonging to the classes inside the tag is run(testng.xml)
- @AfterTest: Will execute after all the test methods belonging to the classes inside the tag have run(testng.xml)
- @BeforeGroups: Will execute before the first test run of that specific group
- @AfterGroups: Will execute after all test methods of that group completes its execution.
- @BeforeClass: Will execute before the first test methods in the current class is invoked.
- @AfterClass: Will execute after all the test methods in the current class have been run.
- @BeforeMethod: Will execute before every @test annotated method
- @AfterMethod: Will execute after every @test annotated method
- @Test: It is a part of a test case
- @Parameters: Used to pass parameters to test methods from testng.xml
- @DataProvider: Pass the different set of data to same test
- @Optional: It defined parameter is not found in testng.xml file. The test method will receive the default value which is specified inside the @optional annotation.

**What is Hard Assert and Soft Assert?**

- In Hard Assert, if any of the assert is getting failed, rest of the lines will not be executed from the particular test method and the test case will marked as failed.
- In Soft assert, even though the asserts are getting failed, rest of the lines will be executed and test case will be marked as pass. In soft assert we have a method called Assert All which is used to highlight the assert failure.