**Test Approach Documentation**

**Introduction to the Problem Statement**

The problem statement involves testing a web-based online calculator. The calculator is accessible through a web page and is expected to perform various arithmetic operations. The testing involves positive, negative, and boundary value scenarios, capturing screenshots, and generating test reports using Extent Reports.

**Test Objectives for the Problem Statement**

1. **Validate Positive Scenarios:** Ensure the calculator produces correct results for valid input combinations.

2. **Handle Negative Scenarios:** Verify that the calculator gracefully handles invalid input scenarios.

3. **Boundary Value Testing:** Test the calculator's behavior at the edge of the input domain.

4. **Generate Test Reports:** Utilize ExtentReports to create detailed and visual test reports.

5. **Capture Screenshots:** Capture screenshots for each test iteration to aid in debugging and reporting.

**Test Scope**

**The scope of the testing includes:**

- Verifying the correctness of arithmetic operations.

- Handling invalid inputs and non-numeric values.

- Checking the calculator's behavior at boundary values.

- Capturing screenshots for every test iteration.

- Logging test results using ExtentReports.

**Test Environment**

- Automation Tool: Selenium WebDriver

- Programming Language: Java

- Test Framework: TestNG

- Reporting Tool: ExtentReports

- Browser: Chrome (WebDriverManager for dynamic driver setup)

- Web Environment: Online calculator accessible via the Google search page

**Approach to Testing the Problem Statement**

**1. Set Up WebDriver:** Initialize the ChromeDriver using WebDriverManager and navigate to the online calculator web page.

**2. Positive Scenario Testing (testOnlineCalculator):**

- Read positive test data from an Excel sheet.

- Iterate through each set of data.

- Enter operands and operators into the calculator.

- Validate the result against the expected result.

- Log test status, capture screenshots, and clear the calculator for the next iteration.

**3. Negative Scenario Testing (negativeTestOnlineCalculator):**

- Read negative test data from an Excel sheet.

- Iterate through each set of data.

- Check for non-numeric values and invalid characters in operands.

- Enter valid data into the calculator.

- Validate the result against the expected result.

- Log test status, capture screenshots, and clear the calculator for the next iteration.

**4. Boundary Value Testing (boundaryValueTestOnlineCalculator):**

- Read boundary value test data from an Excel sheet.

- Iterate through each set of data.

- Enter boundary values into the calculator.

- Validate the result against the expected result.

- Log test status, capture screenshots, and clear the calculator for the next iteration.

**5. Capture Screenshots:**

- Implement a screenshot capture mechanism for both successful and failed test cases.

- Embed screenshots into Extent Reports for visual representation.

**6. Generate Test Reports:**

- Utilize Extent Reports to generate detailed HTML reports.

- Log test results, status, and attach screenshots for reference.

- Flush the Extent Reports instance to ensure all data is captured.

**7. Test Tear Down:**

- After each test method, check if the test failed.

- Log failure status to Extent Reports if applicable.

- Flush the Extent Reports instance after completing all tests.

**Conclusion**

The test approach aims to cover various scenarios, ensuring the online calculator functions correctly, handles invalid inputs gracefully, and produces accurate results at the edge of the input domain. The use of ExtentReports enhances reporting with detailed logs, screenshots, and a clear representation of test results.