TC1: When I click on the input, field should become highlighted

TC2: After adding a number, click on arrow button (resulted in a bug)

TC3: After adding a number, clicking on “Calculate” button should return the good result

TC4: Without adding a number, clicking on “Calculate” should do nothing

Mention here: [...] clicking on “Calculate” should return a warning message informing the user about the input empty value

TC5: Calculate the factorial for 0

TC6: Calculate the factorial for 1

TC7: Calculate the factorial for 1.2

Expected result: An error message should be displayed, informing the user to use only integers

TC8: Calculate the factorial for 10

Expected result: the result should be displayed in the following format: “The factorial of X is: Y”; value displayed should be 3628800

TC9: Calculate the factorial for 9999 (resulted in a bug)

TC10: Calculate the factorial for any 4 digits number (resulted in a bug similar to Bug3)

Expected result: Should return a value or a message informing the user that the result is too big to be processed/displayed

TC11: Calculate the factorial for 170

TC12: Calculate the factorial for 171

Expected results: the result displayed should be “Infinity”

TC13: Calculate the factorial for 191

Expected result: the result displayed should be “Infinity”

TC14: Calculate the factorial for “adsds”

Expected result: an error message should be displayed

TC15: Calculate the factorial for “12/02/1996”

Expected result: an error message should be displayed

TC16: Calculate the factorial for “Infinity”

TC17: Calculate the factorial for “-3”
Expected result: error message should be displayed

TC18: Calculate the factorial for “   “ (white spaces)
TC19: Calculate the factorial for “@-.”

TC20: Click on “Terms and Conditions”
TC21: Click on “Privacy”
TC22: Click on “Qxf2 Services”

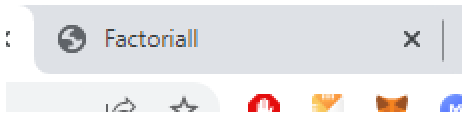
Bug1: *medium prio*

- Arrow next to input field should calculate the factorial
  Steps: 1. Add number in the input field (e.g. 5)
  2. Click on the arrow button (marked in source as input-group-addon)
  Expected res: the system should calculate the fact for 5
  Actual res: Arrow button does nothing

  QA Obs! Arrow button should have a function attached to it, otherwise can be considered a dead button, bringing useless complexity to the UI

Bug2: *low prio*

- Tab title should be “Factorial” or “Factorial Calculator”
  Actual result: tab title is “Factoriall”



Bug3: *medium prio*

- As an user, I should be able to calculate the factorial for 9999
  Actual result: The application is freezing and no results is displayed

  QA Obs: a warning message should be displayed informing the user to calculate the factorial only for smaller than 171

Bug4: *low prio*

- As an user, after adding a wrong number, the error message should disappear
  Steps: 1. Add “asdas” and click on “Calculate”

Bug5: *low prio*

- As an user, when I calculate the fact value for -3, I expect an error message to be displayed
  Actual Result
  2. Error message is displayed and I erase “asdas” and write “10-9”
  3. Click again on “Calculate”

  Expected result: The error message should disappear between usages in order to help the user to understand that displayed is not the same error as earlier.
  QA Obs! For negative values the red highlight of the input field, signaling an error value, is not displayed

  Final note: usually I write steps and expected results for every TC when the time allows, but there are sometimes when due to the time constrictions, I only write the test-scenario and the titles of the test cases in order to better prioritize my work.

Code script, C#, Unit test approach:

Let’s assume this is the factorial function, inside the Factorial class - even if it should have some limitations/validations for [10, 10000] range and also long values instead of int/float. But we are here to test it, so who cares 🙂

```
public float CalculateFactorial(int x)
{
   if (x < 2)
      return 1;

   return x * CalculateFactorial(x-1);
}
```

```
[TestMethod]
public void TenFactorial_resultsIn3MilsAndSo()
{
   Factorial calculator = new Factorial();
   float result = calculator.CalculateFactorial(0);

   Assert.AreEqual(3628800, result);
}
```

```
//We can for the positive case to write some others tests with good values, using equivalence
partitioning, but we know that 170 is the real deal and biggest value for Factorials so we'll test
only for 10, 11, 50, 120, 169

[TestMethod]
public void NineFactorial_resultsShouldNotBeCalculatedInTheApp()
{
   Factorial calculator = new Factorial();
   float result = calculator.CalculateFactorial(9);

   Assert.AreNotEqual(362880, result);

//OR LIKE THIS
//StringAssert.Contains(result.Text, "We cannot calculate for values lower than 10");
//Assuming that CalculateFactorial has exception handlers for these cases (<10 and >10k)
//For a mix of negative and positive testing we are going to use boundary value analysis and
//test negative values such as 9 and 10001, and good values such as 10, 11 and 9999, 10000
//(with the mention for the last two to have infinity result)
}


[TestMethod]
public void One_Seven_ZeroFactorial_resultsInInfinity()
{
   Factorial calculator = new Factorial();
   float result = calculator.CalculateFactorial(170);
   float pos_inf = 1 / 0; //this is the representation of positive infinity in c#

   Assert.AreEqual(pos_inf, result);
//again, assuming that CalculateFactorial has exception handler for 170 case
//All values above 170 should be treated exactly like this, so we can write some other tests for
//9999, 10k
}
```

Code Script, Selenium C# approach

I'm naturally using the Page Object design pattern, although some SOLID principles are not very well applied..it is still the best pattern for our needs so far. I'm going to write only the core for every file (I suppose I don't have to send something that can be compiled, there'll be lots of files and libraries for no reason)

Pagefile.cs:

```
[FindsBy(How = How.Id, Using = "number")]
        private IWebElement InputField { get; set; }

[FindsBy(How = How.Id, Using = "getFactorial")]
private IWebElement Calculate { get; set; }

public void CalculateFactorialOf (string x)
{
            InputField.SendKeys(x);
            Calculate.Click();
}
```

Testfile.cs:

```
        [TestCase("0")]
        [TestCase("asdqwdwd")]
        [TestCase("")]
        [TestCase("@#$%^&")]

        public void NegativeLogin(string value)
        {
            TestFile firstPage = new TestFile(Driver);

                firstPage.Login(user, pass);
                Assert.IsTrue(Driver.PageSource.Contains("Please enter an integer"));

//OR LIKE THIS
// Assert.AreEqual(Driver.FindElement(By.Id("resultDiv")), "Please enter an integer"));

        }
```

Acceptance Criteria for Avax faucet website:

1. As an user of the faucet without a metamask account I want to have a link on the page with a tutorial on how to install and use metamask (in the case I'm using TW or Maiar)
2. As an user of the faucet with metamask, I want to add the test subnet to my wallet with ease - using only a button displayed on the website
3. As an user of the faucet with metamask, I want to add the USDC coin versioned for Avalanche testnet with ease - using a button displayed on the website
4. As an user of the faucet with metamask, I want to request 10 test-USDC per day

5. As an user of the faucet with metamask, I want to able to send back the AVAX I'm not using with ease - having a button which bypasses the transaction directly to my Wallet
6. As an user of the faucet with metamask, I want to connect with my wallet to the website for as long as this session lasts - after that, connection should be canceled
7. As an user of the faucet, I want to add any network to my metamask account using only the "Add subnet to metamask"
8. As an user of the faucet, I want to access the explorer of the selected network with ease - clicking only on one button
9. As an user of the faucet, after successfully sending coins to my wallet, I want to see a link to explorer indicating the transaction and the status.
10. As an user of the faucet, I want to see indicating labels for every section; Also, they should be accompanied by details.