**Objective**

To learn TDD Cycle : Red, Green, Refactor

**Problem Statement**

In mathematics, the Fibonacci numbers are the numbers in the following integer sequence:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144...

<table>
<thead>
<tr>
<th>$F_0$</th>
<th>$F_1$</th>
<th>$F_2$</th>
<th>$F_3$</th>
<th>$F_4$</th>
<th>$F_5$</th>
<th>$F_6$</th>
<th>$F_7$</th>
<th>$F_8$</th>
<th>$F_9$</th>
<th>$F_{10}$</th>
<th>$F_{11}$</th>
<th>$F_{12}$</th>
<th>$F_{13}$</th>
<th>$F_{14}$</th>
<th>$F_{15}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>21</td>
<td>34</td>
<td>55</td>
<td>89</td>
<td>144</td>
<td>233</td>
<td>377</td>
<td>610</td>
</tr>
</tbody>
</table>

**Solution**

By definition, the first two numbers in the Fibonacci sequence are 0 and 1, and each subsequent number is the sum of the previous two.

**Algebraic Equation**

In mathematical terms, the sequence $f(n)$ of Fibonacci numbers is defined by the recurrence relation $f(n) = f(n-1) + f(n-2)$ with seed values $f(0) = 0$, $f(1) = 1$
Visual Representation

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Guidelines

1. Each row in the table is an example. Make each example executable.
2. The final solution should be able to take any random number and calculate the Fibonacci number without any modification to the production code.
Factorial Exercise

Objective

To learn TDD Cycle : Red, Green, Refactor

Problem Statement

The factorial of a non-negative integer n, denoted by n! is the product of all positive integers less than or equal to n.

4! = 4 x 3 x 2 x 1 = 24
7! = 7 x 6 x 5 x 4 x 3 x 2 x 1 = 5040
1! = 1
0! = 1

Solution

You can calculate a factorial from the previous one:

<table>
<thead>
<tr>
<th>n</th>
<th>n!</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2 x 1 = 2 x 1! = 2</td>
</tr>
<tr>
<td>3</td>
<td>3 x 2 x 1 = 3 x 2! = 6</td>
</tr>
<tr>
<td>4</td>
<td>4 x 3 x 2 x 1 = 4 x 3! = 24</td>
</tr>
<tr>
<td>5</td>
<td>5 x 4 x 3 x 2 x 1 = 5 x 4! = 120</td>
</tr>
</tbody>
</table>

Algebraic Equation

n! = n x (n-1)! if n > 0
     1 if n = 0
Guidelines

1. Each row in the solution table is an example. Make each example executable.
2. The final solution should be able to take any random number and calculate the Factorial number without any modification to the production code.
Rock Paper Scissors

Learn the game by playing it here:

Solution:
https://github.com/bparanj/TDD-Workbook/tree/master/rock
Stub Exercise

Objective

To learn how and when to use stub.

Problem

Given the shopping cart class, implement the total method. The shipping calculator component is not available. We know the interface of the shipping calculator to be:

tax(amount, zip) which returns the shipping amount to be charged for the purchase.
Mock Exercise

Objective

To lean how and when to use mock.

Problem

When someone completes a purchase successfully, we need to send them a confirmation email with the receipt for the purchase.