|  |
| --- |
|  |
| TDD |
| The Bowling Kata |
|  |
|  |
|  |

|  |
| --- |
|  |

# 

# Overview

After completing this lab, you should have a better understanding of the red, green, refactor flow in a TDD cycle. We’ll be using Robert C Martin’s “Bowling Kata”.

# Getting Started

1. Open Visual Studio and create a new Console application in the **before** directory of this lab (File -> New Project -> Windows -> Console Application). Name the project **Bowling**.
2. Right-click the Solution in the Solution Explorer Window and select **Add -> New Project**.
3. Select the **Test Project** from the Test node and give the project the name **Bowling.Tests**.
4. **Delete UnitTest1.cs** from the project.

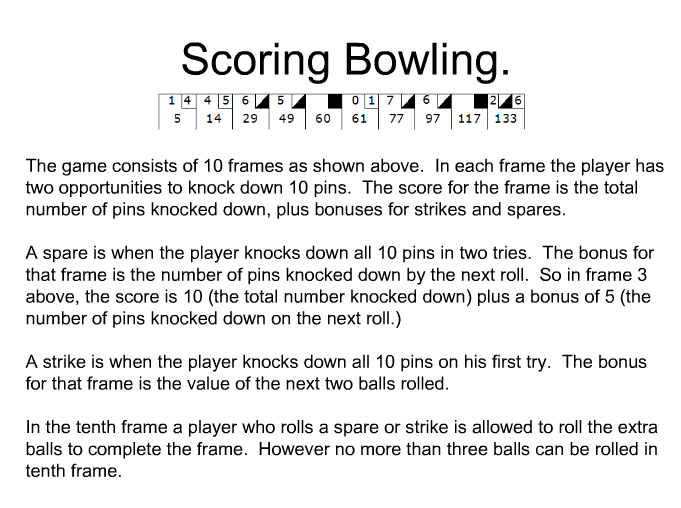
# Part I: The Kata

Write a class that will compute the score for a game of bowling.

There are no requirements to keep track of individual frames or intermediate scores. The API should be designed to accept one ball at a time, in other words for each roll the program will call into the class and tell it how many pins were knocked down. At the end of the game the program will ask the class for the score. The rest of the implementation details are up to you! Just keep these points in mind.

1. Write your tests first. Let the tests drive the design of the class.
2. Start by testing the simplest possible scenario (perhaps a game of gutter balls). Use these simple tests to flesh out the API design.
3. End by testing the hardest scenarios (spares and strikes).
4. Remember Red-Green-Refacotor.
5. Take your time and let the experience seep in. You want to be thinking how you could apply the same approach to your real production code.
6. Write only the “happy day” scenarios for now.
7. Feel free to organize your setup code into a TestInitialize method

If you are unfamiliar with how to score a game of bowling, reference the picture below.



# Part II: The Sad Path

Here are some requirements for illegal or invalid interaction with the scoring class. Write out these requirements into tests, then make the tests pass (if they don’t already).

1. Knocking down less than 0 pins with a roll will throw an InvalidArgumentException.
2. Knocking down more than 10 pins with a roll will throw an InvalidArgumentException.