Scoring Bowling.

1 4	4 5	6 ◢	5 ◢	■	0 1	7 ◢	6 ◢	■	2 ◢ 6
5	14	29	49	60	61	77	97	117	133

The game consists of 10 frames as shown above. In each frame the player has two opportunities to knock down 10 pins. The score for the frame is the total number of pins knocked down, plus bonuses for strikes and spares.

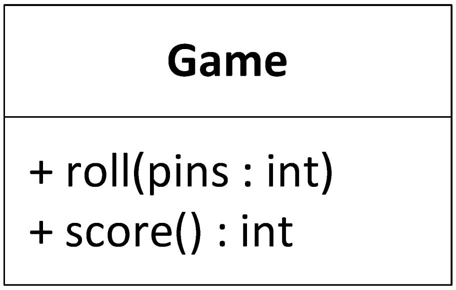
A spare is when the player knocks down all 10 pins in two tries. The bonus for that frame is the number of pins knocked down by the next roll. So in frame 3 above, the score is 10 (the total number knocked down) plus a bonus of 5 (the number of pins knocked down on the next roll.)

A strike is when the player knocks down all 10 pins on his first try. The bonus for that frame is the value of the next two balls rolled.

In the tenth frame a player who rolls a spare or strike is allowed to roll the extra balls to complete the frame. However no more than three balls can be rolled in tenth frame.

So when a strike or spare has been thrown in frame 10, then each pin rolled after that only counts as a bonus for frame 10. In the example a spare is thrown in frame 10. So the score for the last frame is 117 + 10 + 6 (bonus) = 133.

The Requirements.



Write a class named “Game” that has two methods

- roll(pins : int) is called each time the player rolls a ball. The argument is the number of pins knocked down.
- score() : int is called only at the very end of the game. It returns the total score for that game.

Guidelines

- Read the requirements again .
- Use test driven development (TDD). So write your test first and then refactor to pass the test. Start with simple test cases and extend from there.
- You are free to add any fields, create any new objects, methods and properties.
- Keep refactoring while writing your code to make sure it remains simple, maintainable and easy to read.
- The goal of the assessment is not solely to crack the puzzle to return a correct score, but to write good code.