CS100J Spring 2005 Assignment A3 Maintaining a bowling score frame

Submit your solution on the course management system by: 23:59, Wednesday, 1 March.

You may work in groups of 2. But PLEASE form your group by 25 February, well before you submit!

Introduction

The purpose of this assignment is to give you practice with writing method bodies from method specifications. The topic we have chosen, maintaining the score in one frame of a bowling game, requires attention to detail. The logic is just messy enough to cause you to think carefully and proceed slowly.

The files that you hand in are expected to be your own work. Copying of any kind is against the Cornell Code of Academic Integrity. More importantly, such cheating defeats the purpose of the assignment, which is to give you practice in writing programs.

Bowling

A game consists of ten *frames*. In each of the first 9 frames, a player rolls one or two balls:

- A frame starts with ten pins up.
- If the first ball knocks down all ten pins, it is called a strike (written "X"), and a second ball is not thrown. As in golf, the better you are, the less you get to play.
- If the first ball knocks down less than ten pins, a second ball is thrown. If together the two balls knock down ten pins, it is called a spare (written"/").

Here is how the information is recorded for a frame OTHER THAN THE TENTH FRAME:

strike:	X
spare with 7 on the first ball:	7/
less than 10 pins on the two balls, say 4 and 5	45

In the tenth frame, if you get a spare or a strike, you throw three balls; otherwise, just two balls. Whenever all 10 pins are knocked down, they are set up again. So, you can throw three strikes in the tenth frame: XXX; or a spare and a strike, e.g. 5/X; or a spare and less than ten pins, e.g. 5/9; or less than ten pins, e.g. 81. These examples don't include all possibilities.

Here's how one calculates the score in a frame:

- In any frame, if you get less than ten pins on two throws, the score is the number of pins knocked down.
- In any frame but the tenth, if you get a spare, the score is 10 + what you get on the next ball (in the next frame).
- In any frame but the tenth, if you get a strike, the score is 10 + what you get on the next two balls.
- In the tenth frame, the score is the sum of whatever balls you roll.

Below, we show a game for Mary and one for Harry. The first line of each has the person's name together with what they rolled in each of the ten frames. The second line contains the running total score after each frame —this is how you keep a score. But remember that in this assignment, your class will maintain the score only for one frame.

Mary got a strike in every frame and three strikes in the last frame, for a perfect score of 300. Since Mary got a strike in her first frame, her score there is 10 + what she rolled on the next two balls, which were also strikes. Harry got just 9 pins in the first frame. In the second frame, he got a spare, so the score is 10 plus the next ball, which was a strike in the next frame, so his score for the second frame is 20. And so it goes.

Mary	X	X	X	X	X	X	X	X	X	XXX
	30	60	90	120	150	180	210	240	270	300
Harry	45	4/	X	6/	63	X	X	X	X	54
	9	29	49	65	74	104	134	159	178	187

Class Frame

Here is a skeleton of class Frame. It compiles, but it contains only what the user would see —specifications of methods and their headers— together with method bodies that are either empty or contain a simple return statement so that the class can be compiled. You have to add any fields that you need and fill in the bodies of all the methods. You may add extra private methods if they help — make them private because they should not be seen/usable outside the class.

We discuss the various components of class Frame. We urge you to write them and test them thoroughly, in the steps provided below. Don't go on to the next step until you are absolutely sure the first is correct. As you proceed, develop a JUnit testing class FrameTester, in the usual manner. You will have to submit your file FrameTester.java. Before you begin, read the specs of all the methods so that you know what you have to do.

Step 0. Fields, constructors, and getter methods. First, write declarations of whatever instance variables (fields) that you think are necessary. You need to keep track of up to three balls, and you have to know whether it is the tenth frame or not, because that frame is handled differently. Make sure you write definitions of the variables (as comments) next to the declarations.

Second, fill in the body of the first constructor and the getter methods, develop test cases, and test.

Third, fill in the body of the second constructor, develop test cases, and test.

Fourth, fill in the body of the third constructor, develop test cases, and test.

Step 1. Method toString. Write method toString and test it.

Step 2. Write function score, which returns the score for the frame. This one is weird because, as you know, the score in the frame may depend on what happens in the next one or two frames. Therefore, the function has two parameters, which are the next one or two frames. You can test for the tenth frame easily enough, if you set your fields correctly. Here is how you test for the ninth frame: if parameter p1 is not **null** and parameter p2 is **null**, then it is the ninth frame. Program, compile, make up test cases, and test incrementally.

Step 3. Write function record, which produces the information about balls rolled in the frame. The result MUST be THREE characters long, as specified. Pad with blanks, if necessary, to get it three characters long.

This can be a messy function. We suggest that you write a private function to handle the case of the tenth frame, which is totally different. Then, in function card, test whether it is the tenth frame and return its value immediately if it is (by calling the private function).

Suppose it is not the tenth frame. We suggest that you handle one case at a time, returning its value immediately. For example, start off with the case that it is not a spare: if the sum of the two balls is less than 10, say the balls are 5 and 4, then return "54 ". Next, handle the case of a spare, and finally handle the case of a strike. AFTER EACH CASE, put test cases for it in class FrameTester and test it). Do incremental programming, compiling, and testing. That way, you remain under control and you have confidence in what you are doing.

Write and test the case of the tenth frame in the same way.

Step 4. Submit your files Frame.java and FrameTester.java on the CMS by the deadline. Before you do that, make sure your program is well indented. This is easily done by selecting all lines in DrJava and then using menu item Edit->IndentLines. Also, click the Javadoc button is DrJava and take a look at the specification of your class; make sure the method specs are alright.