CPS506 Lab 2

Pharo Smalltalk: Creating classes, adding methods

Preamble

In this lab you will practice creating classes and methods in Pharo through the system browser. You'll practice basic object-oriented principles such as implementing class-side and instance-side methods, and so on.

Lab Warmup

Refer to PowerPoint 3.1 for a short tutorial on creating classes and methods in Pharo. You can also ask your TA for help with this during your lab session.

Lab Description

Start by creating a new package through the System Browser called CPS506, and a new class called **Crag**. Your **Crag** class will implement several different instance methods and one class method related to the simple dice game Crag:

https://en.wikipedia.org/wiki/Crag_(dice_game).

Notice that the order of the dice doesn't matter. Each roll contains three dice, and the values on those dice, regardless of order, determine the strength of the roll.

For each method described below, you must adhere to the constraints provided. For anything that falls outside the scope of the constraints, you have complete freedom. Use whichever Smalltalk language constructs you find most useful. Implement as many helper methods as your heart desires. For an added challenge, implement each method in *as few lines as possible*. Most can be done in one or two.

Your methods may assume the input is correct. The parameter **dice** is guaranteed to be an array containing three integers. No printing whatsoever should occur inside your methods. Remove all prints before submitting.

Class-side methods:

1) **`new: dice`**

   This method creates a new Crag object and assigns the parameter dice to instance variable **`roll`**. Don't forget to define this instance variable in your Crag class!

2) **`surprise`**

   This method randomly generates three integer values between 1 and 6, representing a roll of three dice, and places them in an array. It should then create a new Crag object, assign that array to the instance variable roll, and return the Crag object.

***Hint #1*:** Check out the methods in the **`Random`** class…
***Hint #2*:** This method can call the new: method above after generating the dice roll…

Instance-side methods:

3) **`setRoll: dice`**

   This method should assign the parameter dice to an instance variable called roll.

4) **`getRoll`**

   This method simply returns the instance variable **`roll`**.

5) **`isCrag`**

   If the dice represent a Crag, return **`true`**. Return **`false`** otherwise.

6) **`isThirteen`**

   Return **`true`** if the dice total 13 but are not a Crag. Return **`false`** otherwise.

7) **`isThreeOfAKind`**

   Return **`true`** if the dice are three of a kind, **`false`** otherwise.

8) **`isStraight`**

   This method should return **`true`** if the dice roll contains any of the four straights (low, high, even, odd) described in rules of Crag, and **`false`** otherwise.

9) **`score`**

   This method determines the overall *score* of the roll. The return type should be an integer, whose value is simply the score.

10) **`category`**

    This method determines the *category* of the roll (Low Straight, Odd Straight, Crag, Thirteen, etc.) and returns it as a **`String`**. Format your return string precisely based on the strings below:

`'Crag'`	`'Thirteen'`	`'Three-Of-A-Kind'`
`'Low Straight'`	`'High Straight'`	`'Odd Straight'`
`'Even Straight'`	`'Sixes'`	`'Fives'`
`'Fours'`	`'Threes'`	`'Twos'`
`'Ones'`		

    For the score and category methods, you should consider **all** Crag categories, not just those in the previous methods. For example, sixes, fives, fours, etc.

Testing

You are given a Pharo image along with this lab description. Complete your Crag class and all methods in this image. The Crag class is already created for you, in the package CPS506. You should add the class methods and instance methods indicated above. You are free to implement as many additional helper methods as you want, but all your work should be confined to the Crag class.

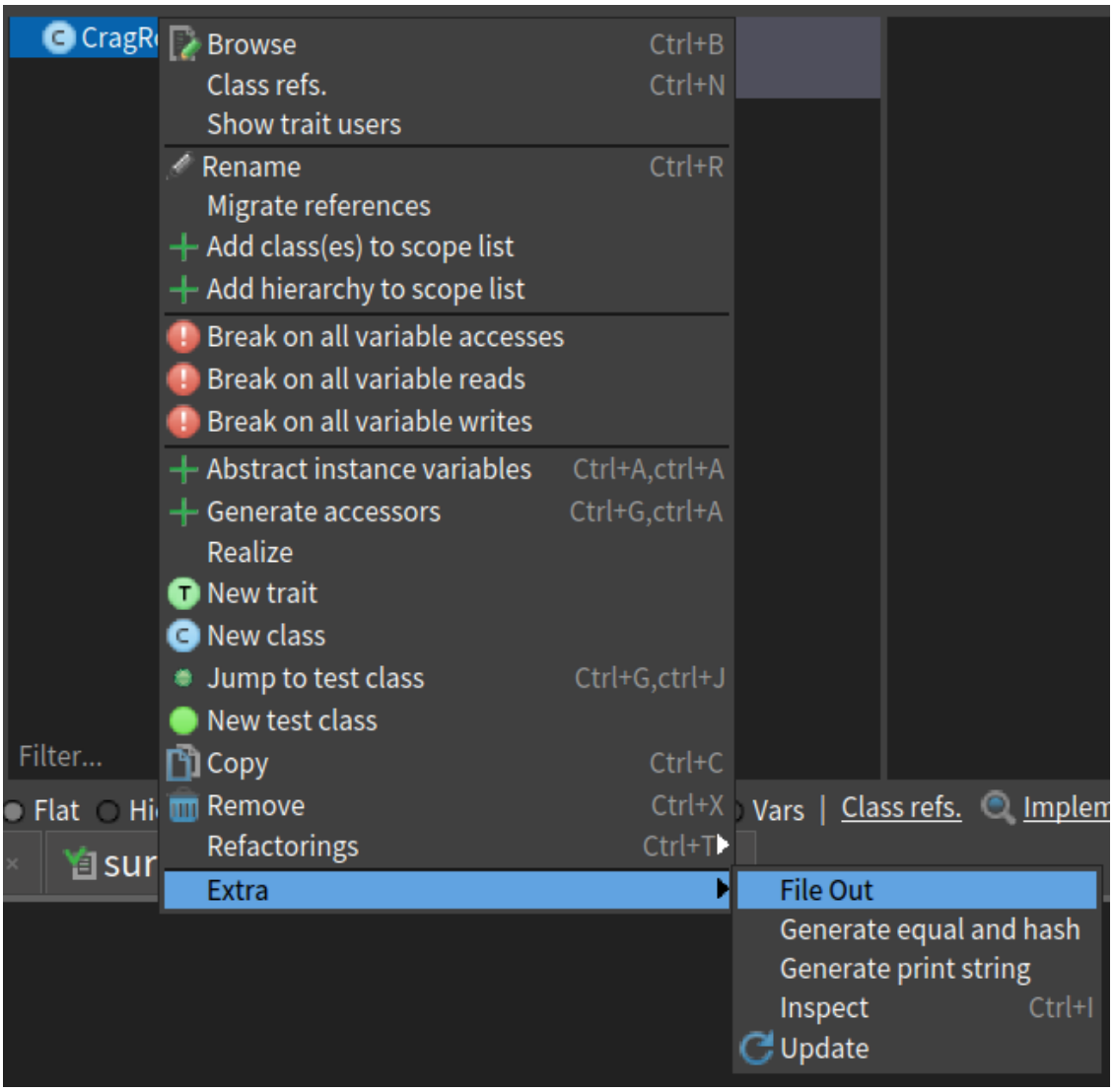
Also included in this image are a set of unit tests, several for each method. These unit tests are in the "CragTest" class, which is also in the CPS506 package. Run these unit tests to validate your Crag methods. To do this, right-click on the CragTest class in the system browser and select "Run tests". When they all turn green, you're good to go.

Your mark for this lab will be based on passing the included unit tests. Do NOT modify the unit test code to try and cheat the system. Your code will be marked using a fresh image with unmodified unit tests. Additionally, do NOT hard-code for the unit tests. Doing so will result in a mark of zero for the lab.

Submission

Labs are to be completed and submitted *individually*. Submit the following item(s) on D2L, under the submission for Lab #2:

1) The plain text Crag.st file containing the source code of your Crag class. Obtain this file by right clicking the class name and selecting "Extra->File Out". It will be written in the same folder as your Pharo image. See the screenshot below.


CragR
Browse Ctrl+B
Class refs. Ctrl+N
Show trait users
Rename Ctrl+R
Migrate references
Add class(es) to scope list
Add hierarchy to scope list
Break on all variable accesses
Break on all variable reads
Break on all variable writes
Abstract instance variables Ctrl+A,ctrl+A
Generate accessors Ctrl+G,ctrl+A
Realize
New trait
New class
Jump to test class Ctrl+G,ctrl+J
New test class
Copy Ctrl+C
Remove Ctrl+X
Refactorings Ctrl+T
Extra
File Out
Generate equal and hash
Generate print string
Inspect Ctrl+I
Update
Filter...
Flat
Vars | Class refs.
Implem