Prof. Dr. Oscar Nierstrasz
Pascal Gadient, Pooja Rani

Solution
Assignment 04 — 10/10/2018 – v1.0a
Smalltalk: Reflection

Please submit this exercise by mail to sma@list.inf.unibe.ch before 17 October 2018, 10:15am.

Exercise 1 - Hierarchy traversal (2 Points)

Write a method that finds the class with the longest inheritance chain among all Smalltalk classes in the Pharo programming environment.
Hint: To access all classes of Smalltalk, you can use `Smalltalk allClasses`. **Answer:**

```
(((Smalltalk allClasses collect:
  [:eachClass | eachClass -> eachClass classDepth]) sorted:
    [ :a :b | a value > b value ]) asOrderedDictionary) keys first
```

`keys first` *will point to the class with the longest inheritance chain,* i.e. *the class* `QuestionWithoutCancelDialogWindow` *at the end of a chain with 12 elements in the Pharo 6.1 stock image. Please note that there exist three more classes of the same hierarchy level:* `PasswordDialogWindow`, `IceGitHubPullRequestBrowser`, *and* `CustomQuestionDialogWindow`.

Exercise 2 - Method overrides (2 Points)

Write a method to find all methods that override an abstract method in the Pharo system. **Answer:**

```
(SystemNavigation default allMethods select:  #isAbstract) flatCollect:
  [ :m | ((m methodClass allSubclasses flatCollect:  #methods) select:
    [ :n | m selector = n selector ]) reject:  #isAbstract ]
```

Exercise 3 - Query methods (2 Points)

Write a method that finds all classes in the Pharo environment with at least one query method.
Hint: Query methods test a property of an object. In Pharo these methods are prefixed with `is`, `was` *or* `will`. **Answer:**

```
SystemNavigation default allClasses select:
  [ :class | class methodDict keys anySatisfy:
    [ :sel | ('is*' match:  sel) |
    ('was*' match:  sel) | ('will*' match:  sel)]
  ].
```

Prof. Dr. Oscar Nierstrasz
Pascal Gadient, Pooja Rani

Exercise 4 - Root methods (2 Points)

Find all *root methods* in the off-the-shelf Pharo image.
Hint: A "root method" is a method whose selector has been implemented in a class, such that the super classes of that class does not understand it. **Answer:**

```
introducedMethods := [ :class | class superclass
  ifNil:  [ class methods ]
  ifNotNil:  [ class methods select:
    [ :met | (class canUnderstand:  met selector) &
    (class superclass canUnderstand:  met selector) not ]]].

SystemNavigation default allClasses flatCollect:
  [:cl | introducedMethods value:  cl].
```

Exercise 5 - Dynamic coding (2 Points)

Based on the code used in assignment 02 (downloadable from here), do the following exercise:

Step 1: Redefine the method `Call doesNotUnderstand:  aMessage` and within this method dynamically add two elements to the class `Call`, namely the instance variable `numberOfArguments`, and the method described below.

```
Call>>#numberOfArguments
    numberOfArguments := args size.
    ↑ numberOfArguments
```

Step 2: Execute the code

```
(CallGraph fromFile: 'Calls.txt') calls collect: [ :each | each
   numberOfArguments]
```

and ensure it prints the number of arguments for every call in the call graph without raising a doesNotUnderstand error. **Answer:**

```
doesNotUnderstand:  aMessage
|messageName|
messageName := aMessage selector asString.
messageName = 'numberOfArguments'
  ifTrue:  [ (self class allInstVarNames includes:  'numberOfArguments')
    ifFalse:  [ self class addInstVarNamed:  'numberOfArguments'].

    self class compile:
    messageName, String cr,
    messageName, ' := args size.', String cr, '^', messageName, '.'.

^ aMessage sendTo:  self.  ].
```