Assignment 02 — 23.09.2020 – v1.0
Smalltalk: A Reflective Language

Please submit this exercise by mail to pascal.gadient@inf.unibe.ch before 30 September 2020, 10:15am.

For this assignment you have to download and install Glamorous Toolkit (GT) from here.

Exercise 1 - Nature of Smalltalk and GT (5 pts)

a) What do you have to consider when you use a live system for software development? In other words, name one major threat that cannot arise in traditional software development, *e.g.* with Java, C++, etc.?

b) What is a message in Smalltalk?

c) What is a block in Smalltalk?

d) How do Smalltalk, Pharo, and GT relate to each other?

e) What are counterparts of GT tools (Playground, Coder, Git, Monitor, ExamplesExplorer, Transcript, Morphic World, Spotter) in your favorite development environment?

Please continue reading on the next page.

Exercise 2 - Object inspection (4 pts)

a) Explain the difference between a `String` and a `Symbol` object in Smalltalk. Why is this differentiation important?
Hint: The execution of the code below will reveal differences.

```
('HeySmalltalker') == 'HeySmalltalker'.
('HeySmalltalker') asSymbol == #HeySmalltalker.
('Hey','Smalltalker') == 'HeySmalltalker'.
('Hey','Smalltalker') asSymbol == #HeySmalltalker.
```

b) Write the equivalent of the following piece of code in Smalltalk as a block, and execute it with the values <38, 44>, <65, 48>, and <48, 48>.

Hint: Use the transcript tool (one line example below) as replacement for the `print` *method.*
`Transcript show: 'your output here'`

```
    int scoreOfPlayerA, scoreOfPlayerB;
    if(scoreOfPlayerA > scoreOfPlayerB)
        print "Player A Won"
    else if(scoreOfPlayerA < scoreOfPlayerB)
        print "Player B Won";
    else
        print "Match is declared as draw";
```

c) How many classes in GT implement the message `includes: anObject`?
How many messages in GT use that particular message?
Note: For this task, you are supposed to use a fresh and unaltered copy of GT with no changes of yours.

d) Which message in GT can be sent to a class to find all its subclasses?

Please continue reading on the next page.

Exercise 3 - CallGraph (1 pt)

Find the top ten most frequently invoked methods in the provided CallGraph representation.

In order to perform this exercise, you must (i) execute the statement below to retrieve the “CallGraph Demo” in GT, and (ii) you have to download the `Calls.txt` file from here and store it in the same folder as the GT image file.

```
Metacello new baseline:  'SMAForGt';
repository:  'github://onierstrasz/sma-examples/src'; load.
```

Hint 1: You can find some examples for accessing the CallGraph in the “CallGraph Demo”.

Hint 2: Don’t forget to submit your code snippet and your results.