University College Dublin
An Coláiste Ollscoile, Baile Átha Cliath

SEMESTER I EXAMINATIONS

ACADEMIC YEAR 2018/2019

COMP 47530

Exploring Ruby (Mixed Delivery)

Assoc. Prof. V. Dimitrova

Prof. P. Cunningham

Prof. M. Keane *

Time Allowed: 2 Hours

Instructions for Candidates

Answer any FIVE questions.

All questions carry equal marks.

Total marks available 100.

Use of calculators is prohibited.

Instructions for Invigilators

Use of calculators is prohibited.

1. Create two methods – `seq_gen_a` and `seq_gen_b` -- each of which will take a number, *n* (which is > 2), and generate an array of four elements, whose first element is *n* and next three elements are three numbers in a sequence that doubles the previous number and takes 3 from it; such that, (i) `seq_gen_a` generates the sequence using iteration, and (ii) `seq_gen_b` generates the sequence using recursion.

   For example, given the number `5`, both of these methods will output:

   ```
   [5, 7, 11, 19]
   ```

   though, obviously, they will achieve this output in different ways.

2. Define a class called `TeamGame` (with three attributes) and a subclass of it called `RugbyGame` (with five attributes, one of which is `name`). Also, define a module called `Labeller` with a method called `label`, that adds the string `" is a sort of game"` to the value of the `name` attribute of any object on which it is invoked.

   Create three methods for both of the classes, such that the `RugbyGame` sub-class inherits, at least, one method from its superclass, `TeamGame`.

   Create a mixin that uses the `Labeller` module in both of these classes so that any object instance of `RugbyGame` will have its name-value modified when invoked with the `label` method. What is a mixin and why does Ruby use them?

3. Write an iterative method (using `each, collect or select)` – called `pluralise` – that will take an array of symbols (of any arbitrary length), such as:

   ```
   [:alpha, :beta, :kappa, :phi]
   ```

   that will add `e` to the symbol if it ends in an `a` and `s` to the end of the symbol if it ends in an `i`. So, for the above array, the method should return the modified array:

   ```
   [:alphae, :betae, :kappae, :phis]
   ```

   Now, define a method – called `pluralise_sub` – that does the same thing using `sub` or `gsub`.

   Now define a method – called `number_off` – that will return the array as an array showing the number of characters in each symbol-element of the array; for example, when dealing with the above original array it should return:

   ```
   [5, 4, 5, 3]
   ```

4. Write a short explanatory paragraph on any ***four*** of the following, using appropriate examples: polymorphism, data abstraction, duck typing, modularity, inheritance in OOP.

5. Ruby on Rails makes use of the Model-View-Controller architecture pattern to organize the development of web-based applications. What are models, views and controllers? Write a short explanatory paragraph on each. Give three reasons why it might be a good idea to divide up web-based applications in this way.

6. Describe what Ruby does during *method lookup*, when an object calls a method (be it an instance or class method), how it searches for the method's definition and the conditions under which it eventually returns a `method_missing` error.

7. What do the following evaluate to in Ruby:

```
i.    print "hammy hamster"
ii.   a = "foo"; p a.to_sym
iii.  ["1","2", 3].instance_of?(String)
iv.   ["a","b,"c"].instance_of?(Array)
v.    class NoClass
      end
      p NoClass.new
vi.   [1,2,3].each
vii.  ["a","b","c"].collect{|item| puts item + "a"}
viii. ["a1","2","c33"].select {|item| item.size == 2}
ix.   [[2,3],[[[3]],[4,5]]].length
x.    [1,2,[3,4],4,2,[[3,[6,2,1]]],145,4,3,2].flatten
xi.   Float.new
xii.  "fooble".concat("doodle")
xiii. ["fooble "].concat(["doodle"])
xiv.  ["fooble"] << ["doodle"]
xv.   "fooblinggg".chomp.chop.chop
xvi. baDDarT.upcase
xvii.   "apples_oranges_lemons".split(/ /)
xviii.          "1234" <=> "12345"
xix.    Regexp.new("eeeeeeek")
xx.     [6,3,2,1].inject{|x,y| x / y}
```