University College Dublin
An Coláiste Ollscoile, Baile Átha Cliath

AUTUMN TRIMESTER EXAMINATIONS

ACADEMIC YEAR 2019/2020

COMP47530
Exploring Ruby (Mixed Delivery)

Prof. Simon Thompson

Dr. Chris Bleakley

Prof. Mark 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

Student Number

Seat Number

Instructions for Invigilators
Use of calculators is prohibited

1. In Ruby, methods can be **public**, **private** or **protected**. Explain the exact meaning of these terms in Ruby [10 marks]. Illustrate your answer with an example of each; that is, define some sample methods within a class and show what happens when they are called in different ways [10 marks].

2. Write a method that tests to see if a number is a prime, as in:

   ```
   13.is_prime?  =>  true
   ```

   and then define two methods that use this `is_prime?` method to find the first 20 primes, checking each number counting up from 1, giving the following outputs:

   ```
   This is a prime: 2
   This is a prime: 3
   This is a prime: 5
   This is a prime: 7
   This is a prime: 11
   This is a prime: 13
   This is a prime: 17
   This is a prime: 19
   This is a prime: 23
   This is a prime: 29
   This is a prime: 31
   This is a prime: 37
   This is a prime: 41
   This is a prime: 43
   This is a prime: 47
   This is a prime: 53
   This is a prime: 59
   This is a prime: 61
   This is a prime: 67
   This is a prime: 71
   ```

   Of these two methods there should be one called (i) `find_primes1` that uses iteration [10 marks] and (ii) `find_primes2` that uses recursion [10 marks].

   (Hint: the modulo operator in Ruby is `%`, as follows: `10 % 2 => 0`)

3. 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 [15 marks]. Give three reasons why it might be a good idea to divide up web-based applications in this way [5 marks].

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 marks for each part].

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

   ```
   [:change, :kiss, :kick, :please]
   ```

   and produce the appropriate past-tense form for these regular verbs [5 marks]. So, for the above array, the method should return the array:

   ```
   [:changed, :kissed, :kicked, :pleased]
   ```

   Now, define a method – called `past_tense_sub` – that does the same thing using **`sub`** or **`gsub`** [5 marks].

   Now define a method – called `count_letters` – that will return the array as an array showing the number of letters in each symbol-element of the array [5 marks]; for example, dealing with the above original array it should return:

   ```
   [6, 4, 4, 6]
   ```

   Is it good practice to use symbols in this way? Briefly list some of the uses symbols are put to in Ruby [5 marks].

6. Describe what the implementation of Ruby does during *method lookup*, when an object calls a method (be it an instance or class method), how the implementation searches for the method's definition and the conditions under which it eventually returns a `method_missing` error [20 marks].

7. What do the following evaluate to in Ruby [1 mark for each part]:

```
i.     puts "hammy hamster"
ii.    ["1","2", 3].instance_of?(String)
iii.   ["a","b,"c"].instance_of?(Array)
iv.    class NewClass ; end
v.     [1,2,3].each
vi.    ["a","b","c"].collect{|item| puts item + “a”}
vii.   ["a1","2","c33"].select {|item| item.size == 3}
viii.  [[[2,3]],[[[3]],[4,5]]].length
ix.    [1,2,[3,4],4,2,[[3,[6,2,1]]],145,4,3,2].flatten
x.     bar = "foo"; p bar.to_sym
xi.    "weather".concat("forecast")
xii.   ["this"].concat(["and, that"])
xiii.  ["tick, tack"] << ["toe"]
xiv.   "sabellantderrree".chomp.chop.chop.chop
xv.    SmeaLugg.upcase
xvi.   "apples oranges lemons".split(/_/)
xvii.  "4567" <=> "45678"
xviii. Regexp.new(" ")
xix.   [1,2,3,4,5].inject{|x,y| x + y}
xx.    a = 1; b = a; p a
```