Laboratory Work #3

on PPC

“Distributed programming”

Done by: Ursu Dumitru
Verified by: Ciorbă Dumitru

Chișinău, 2014

Task:

Se cere un program care sa creeze 15 threaduri 'studenti'. Programul principal (adica profesorul) are o lista de intrebari (adica un vector cu 15 valori numerice arbitrare). Studenti intra pe rand la examen si primesc cate o intrebare la care trebuie sa raspunda. Raspunsul corect este dat de o functie f(k) pe care doar profesorul o cunoaste (pentru ca studentii nu au fost la cursuri). Deci studentul examinat va raspunde cu o valoare aleatoare, incercand sa ghiceasca si primind o nota in functie de cat de aproape a fost de raspunsul corect. In final, dupa ce toti studenti au fost examinati, profesorul va chema in sala primii 3 studenti cu cele mai bune note pentru a le nota numele (adica threadurile studenti vor afisa pe ecran numele/numarul lor, plini de bucurie).

Domain of the problem:

We have lots of threads that need to be syncronized: it's time to use a mutex and a Condition Variable! ConditionVariable objects augment class Mutex. Using condition variables, it is possible to suspend while in the middle of a critical section until a resource becomes available.

Source code analysis:

```ruby
require 'thread'

def examen(student, true_answer)
  # the student and the "distance" of it's guess are returned
  [student[0], (student[1] - true_answer).abs]
end

def f(k)
  k * k + 20
end

questions = Array.new 15
questions.map! { rand 100 }
```

We store some random numbers as the questions...

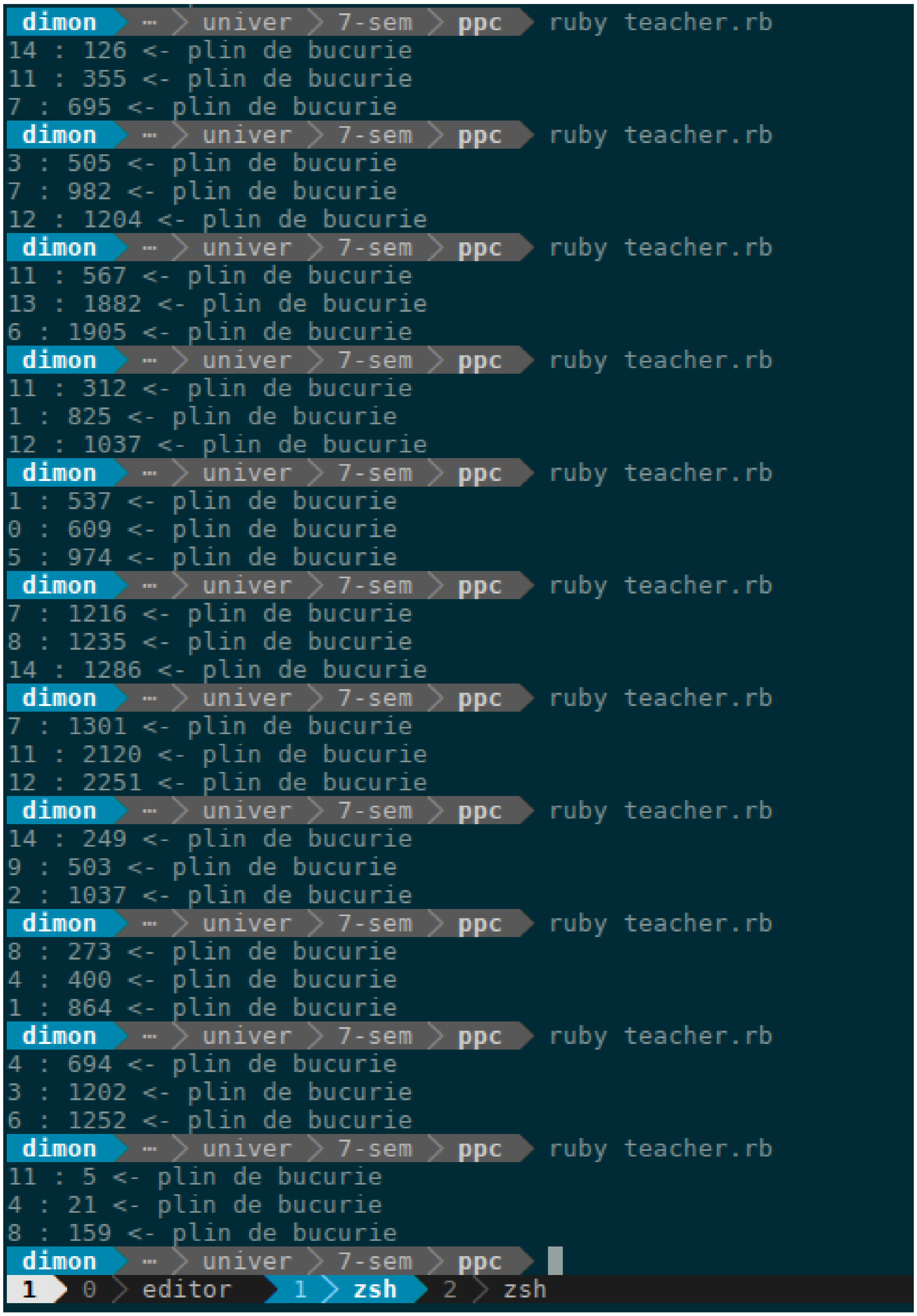
```ruby
mutex = Mutex.new
teacher = ConditionVariable.new
$answers = []
students = []
15.times do |n|
  students << Thread.new(n) {
    # one student at a time, please
    mutex.synchronize {
      # wait the teacher
      teacher.wait(mutex)
      $answers << [n, rand(9000)]
    }
  }
end
```
To find out who are the best students, we make a register, where we store the number of the student, and how close he was to guess the answer.

```
the_register = []
15.times do |n|
  # the teacher gives the signal that it's done with the student
  teacher.signal
end
teacher.broadcast

students.each(&:join)
$answers.each {|answer| number = rand 15; the_register << examen(answer,
f(questions[number])) }

# sorting the registry
the_register.sort_by!{|e| e[1]}

# and, the best students are...
the_register[0..2].each do |best|
  puts "#{best[0]} : #{best[1]} <- plin de bucurie"
end
```



```
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
14 : 126 <- plin de bucurie
11 : 355 <- plin de bucurie
7 : 695 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
3 : 505 <- plin de bucurie
7 : 982 <- plin de bucurie
12 : 1204 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
11 : 567 <- plin de bucurie
13 : 1882 <- plin de bucurie
6 : 1905 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
11 : 312 <- plin de bucurie
1 : 825 <- plin de bucurie
12 : 1037 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
1 : 537 <- plin de bucurie
0 : 609 <- plin de bucurie
5 : 974 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
7 : 1216 <- plin de bucurie
8 : 1235 <- plin de bucurie
14 : 1286 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
7 : 1301 <- plin de bucurie
11 : 2120 <- plin de bucurie
12 : 2251 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
14 : 249 <- plin de bucurie
9 : 503 <- plin de bucurie
2 : 1037 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
8 : 273 <- plin de bucurie
4 : 400 <- plin de bucurie
1 : 864 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
4 : 694 <- plin de bucurie
3 : 1202 <- plin de bucurie
6 : 1252 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉 ruby teacher.rb
11 : 5 <- plin de bucurie
4 : 21 <- plin de bucurie
8 : 159 <- plin de bucurie
dimon  … 〉 univer 〉 7-sem 〉 ppc 〉
1 〉 0 〉 editor 〉 1 〉 zsh 〉 2 〉 zsh
```

Conclusion:

Doing this laboratory work I've learned how to manage an array of workers (students), and how to make sure they work in the fashion I need.

Bibliography:

1. http://ruby-doc.org/stdlib-2.0.0/libdoc/thread/rdoc/ConditionVariable.html
2. http://www.ruby-doc.org/core-2.0.0/Thread.html