CSc 372

Comparative Programming Languages

33 : Ruby — Exercises

Department of Computer Science
University of Arizona

collberg@gmail.com

Christian Collberg

Traversing an array

- Write a method which returns an array of the odd-indexed elements $a_1, a_3, \ldots$ for an array a.
- Hint: Use `for i in ...`.

```
def oddElements0(l)
   ...
end

puts oddElements0([1,2,3,4,5,6])
```

Traversing an array

- Write a method which returns an array of the odd-indexed elements $a_1, a_3, \ldots$ for an array a:
- Hint: Use `each_with_index()`.

```
def oddElements1(l)
   ...
end

puts oddElements1([1,2,3,4,5,6])
```

Traversing an array

- Write a method which yields the odd-indexed elements $a_1, a_3, \ldots$ for an array a:

```
def oddElements2(l)
   ...
end

oddElements2([1,2,3,4,5,6]) do |x|
   puts x
end
```

Traversing an array

- Write a method which yields the odd-indexed elements $a_1, a_3, \ldots$ for an array a if the user supplied a block to the method, and which returns an array of the results otherwise.
- Hint: the method `block_given?` returns `true` if a method was invoked with a block.

```
def oddElements3(l)
 ...
end

puts oddElements3([1,2,3,4,5,6])
oddElements3([1,2,3,4,5,6]) do |x|
   puts x
end
```

Traversing a file

- Write a method that reads in a file line by line and prints the file, with line numbers, on standard output.
- Hint: use a while-loop and `gets`.

```
/* Duckburg sites: */

Town hall
Scrooge's money bin
Gyro Gearloose's lab
```

⇒

```
1
2 /* Duckburg sites: */
3
4 Town hall
5 Scrooge's money bin
6 Gyro Gearloose's lab
```

Traversing a file

- Write a method that reads in a file line by line and prints the file, with line numbers, on standard output.
- Hint: use `each()` to read lines from the file.

Searching a file

- Write a method `sameword(file)` which searches through a file for any potential word duplications such as ``` ``the the'' ```.
- How would you extend this to search for duplications that occured across two lines (``` ``...the\nthe...'' ```)?

```
def sameword (file)
   ...
end

sameword("sametest")
```

Searching a file — Example

```
"there there", she said!
Oh the the humanity!
It was that that made him furious!
"was wasn't, whatever", she said.
Oh, I can't believe the
the humanity!
```

$\Downarrow$

```
Found "there there" on line 1
Found "the the" on line 2
Found "that that" on line 3
```

Searching through the password file

- Write a method `passwd(user,field)` which searches through the `"/etc/passwd"` file on a unix system for a particular user entry, and returns field number `field`.

```
def passwd (user,field)
   ...
end

puts passwd("root",4)
```

$\Downarrow$

```
System Administrator
```

Searching through the password file...

- You should ignore any line whose first non-whitespace character is #.
- Fields in the password file are separated by a colon:

```
##
# User Database
#
nobody:*:-2:-2:Unprivileged User:/:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
daemon:*:1:1:System Services:/var/root:/usr/bin/false
lp:*:26:26:Printing Services:/var/spool/cups:/usr/bin/false
postfix:*:27:27:Postfix User:/var/spool/postfix:/usr/bin/false
```

COBOL

- In COBOL, you write `"ADD 11 TO x GIVING y"` instead of `y = x + 11` as you might in modern languages.
- Write a method `COBOL(s)` which translates a COBOL expression (a string) into the equivalent Ruby.
- Hint: Use an attrocious regular expression with backslash-sequences.

```
def COBOL(s)
   ...
end
puts COBOL("ADD 11 TO x GIVING y")
puts COBOL("MULTIPLY 2 TO y GIVING y")
```

COBOL...

⇓

```
y = x + 11
y = y * 2
```

- Note, that now we can use Ruby's `eval` method to evaluate these expressions:

```
x = 10
y = 0
eval COBOL("ADD 11 TO x GIVING y")
puts y
eval COBOL("MULTIPLY 2 TO y GIVING y")
puts y
```

⇓

```
21
42
```

wc

- Write a replacement for the unix command `wc` which prints out the length (in number of lines, words, and characters) of a file.

```
def wc (file)
    ...
end

wc("/etc/passwd")
```

⇓

```
36 137 1861
```

Word Count

- Write a method which reads words from a file (a word consists of letters and apostrophes) and prints out how many times each word occurs in the file. The word list should be sorted.

```
def words (file)
   ...
end

words("sametest")
```

⇓

```
I            1
It           1
Oh           2
```

Java generator

- Write a method `java(klass, arg1, arg2,...)` which generates a Java class. Each of the `args` is a string of the form `"name/type"` describing the fields the class should have.
- Hint: use variable length argument lists and *here*-documents.

```
def java(klass, *fields)
   ...
end

java("Point", "x/int", "y/int")
```

Java generator…

- The call java("Point", "x/int", "y/int") should write the class below to the file Point.java:

```
public class Point {
   int x;
   int y;
   public Point (int x,int y) {
      this.x = x;
      this.y = y;
   }
   public void setx(int x) {this.x=x;}
   public int getx() {return x;}
   public void sety(int y) {this.y=y;}
   public int gety() {return y;}
}
```

If it quacks like an alien…

Chronicle / Carlos Avila Gonzalez

…an alien has now been found in the stomach of a duck. That, at least, is the conclusion reached by workers at the International Bird Rescue Research Center in Cordelia (Solano County) when they viewed an X-ray image they took of a sick mallard. Right there, in the duck's ventriculus, or gizzard, is the shocking image of a grimacing, bald-headed being. …when an autopsy was performed …the alien had mysteriously disappeared.
From: http://www.sfgate.com/cgi-bin/article.cgi?f=/c/a/2006/05/26/DUCK.TMP