CIT 483/583 Lab – Regular Expressions

Fall 2018

**Instructions**

Open and save this file in any MS word-compatible format as Lab04\_*Firstname*\_*Lastname*.<ext> and place your answers in that document. Do a **Save-As** and retain all of my content. Keep the document safe in case your submission fails, or you discover an error prior to the due date and wish to re-submit. Submit your document to the Lab04 dropbox in Canvas. The due date and any other pertinent information are noted in the Canvas item.

This lab should be completed on students.cs.nku.edu as a reference implementation and as a means of testing your answers. It can be done using IRB only, or you may create .rb files to test some or all of your code, but do not submit .rb files (programs).

**Place the results of the command or other answers in or immediately following each question and make sure your answers stand out from the questions by using a different font color.**

1. For the following, we will use the dictionary file located on students.cs.nku.edu at /usr/share/dict/american-english. Write and test a regular expression that will find the words displayed; show the expression and the actual output--even if it doesn't work 100%. Try to use Ruby RegExp techniques shown in the slides and reading, e.g., groups, repetition, etc.

For example, if the output requested are the words “abuzz" and "adz”, you might write the regexp /^a.+z$/ or /^a.\*z$/.

To test it in IRB, it will be more readable with variables for file and re:

>> re = /^a.+z$/

>> file = "/usr/share/dict/american-english"

>> File.open(file).each\_line { |x| puts x if x =~ re } # fits on one line

abuzz

adz

=> #<File:/usr/share/dict/american-english>

Or you may test it using the Ruby interpreter and a one-liner from the command prompt:

ruby -e ' File.open("/usr/share/dict/american-english").each\_line { |x| puts x if x =~ /^a.+z$/ }'

abuzz

adz

(a) Guadalajara

Mahabharata

abracadabra

**/^[A-Za-z]{7}[j,r,a][a,b][r,t]a$/**

**/^.\*a.+a.+a.+a.+a$/**

**/^.\*a(.+a){4}$/**

**/^.\*a(.\*a){4}$/**

(b) onomatopoeia

pharmacopoeia

queue

sequoia

**/^[a-z]+[aeiou]{4}$/**

(c) counterrevolutionaries

electroencephalograms

electroencephalograph

electroencephalographs

**/^[a-z]{19}[^']\*[s,h]$/ # but not /^[a-z]{19}.\*[s,h]$/**

**/^[a-z]{15}o[a-z]{5,6}$/**

**/^[a-z]{21,22}$/**

**/[a-z]{21}$/**

(d) FNMA

HSBC

NASCAR

NORAD

OHSA

SARS

UCLA

**/^[A-Z]{4,6}$/**

1. Use a regular expression to match the following. As before, to test your pattern, you can use a line (in IRB or using the Ruby interpreter) similar to the previous question. Show the results even if they do not return all of the lines you expect.

HINTS:

* Before you try the expression, take a few minutes and cat the file to understand it's structure.
* You may also use the negated version of =~ which is **!~** to check if a RegExp does not match the line.

For example, if I were looking for lines in /etc/password in which the GECOS field (the 5th field) begins with a capital letter, I might use the line beginning anchor and a group that ends with : 4 times to make sure I am checking the proper field.

$ ruby -e 'File.open("/etc/passwd").each\_line{ |x| puts x if x.chomp =~ /^([^:]\*:){4}[A-Z]+/}'

list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin

gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin

coiadmin:x:1000:1000:COIadmin,,,:/home/coiadmin:/bin/bash

The general form will be one of two options, the first if you prefer IRB, the second if you prefer a command-line one-liner.

>> File.open("/etc/passwd").each\_line { |x| puts x if x.chomp =~ / <a regexp> / }

$ ruby -e 'File.open("/etc/passwd").each\_line { |x| puts x if x.chomp =~ /<a regexp>/ }'

1. Write a regular expression that will only match the lines from /etc/passwd of users whose home directory is under /home.

File.open("/etc/passwd").each\_line{ |x| puts x if x.chomp =~ **/([^:]\*:){5}\/home/**}

NOTE that you can streamline testing by using re again:

re = /([^:]\*:){5}\/home/

File.open("/etc/passwd").each\_line{ |x| puts x if x.chomp =~ re }

1. Write a regular expression that will only match the lines from /etc/passwd of users whose shell is **not** /bin/false.

File.open("/etc/passwd").each\_line{ |x| puts x if x.chomp =~ **/(([^:]\*:){6})(?!\/bin\/false)/**}

File.open("/etc/passwd").each\_line{ |x| puts x if x.chomp =~ **/^(.\*:){6}(?!\/bin\/false)/**}

File.open("/etc/passwd").each\_line {|x| puts x if x.chomp =~ **/[^\/false]$/ }**

**File.open("/etc/passwd").each\_line {|x| puts x if x.chomp !~ /(.\*:){6}\/bin\/false$/ }'**

1. Write a regular expression that will only match the lines from /etc/group of groups that have no (supplementary) members.

File.open("/etc/group").each\_line{ |x| puts x if x.chomp =~ **/:$/** }

1. Write a regular expression that when matched against a line from /etc/passwd will "capture" (e.g., using parentheses) the user's home directory.

File.open("/etc/passwd").each\_line{ |x| puts **/(([^:]\*:){5})([^:]\*)/**.match(x.chomp)[3] }

1. Write a regular expression that when matched against a line from /etc/fstab will "capture" the device path, the mount point and the type of file system.

File.open("/etc/fstab").each\_line {|x| if x[0] != "#" then puts **/^([^\s]+)[\s]+([^\s]+)[\s]+([^\s]+)[\s]+/**.match(x.chomp)[0]; end }

# first group is anything other than white space or #

File.open("/etc/fstab").each\_line do |x|

md = **/^([^#\s]+)[\s]+([^\s]+)[\s]+([^\s]+)[\s]+/**.match(x.chomp)

if md

puts "#{md[1]} #{md[2]} #{md[3]}"

end

end

# more explicit version for first group

File.open("/etc/fstab").each\_line do |x|

md = **/^([A-Za-z0-9\/\-\=\_]+)[\s]+([^\s]+)[\s]+([^\s]+)[\s]+/**.match(x.chomp)

if md

puts "#{md[1]} #{md[2]} #{md[3]}"

end

end