Lecture

whereever you put the require/require\_relative in your file, that is where itll run the file. Example. file.rb puts “bob” and other\_file.rb puts Hello

if you go in file.rb ---- puts “bob”

require\_relative ‘other\_file.rb’

It will print bob

Hello

Regex groups

M2M5C5

asdfsas

@M3G44

^[A-Z]\d[A-Z] \d[A-Z]\d$

or [a-zA-Z]\d[a-zA-Z] \d[a-zA-Z]\d

a regex group is just putting brackets around parts of your regex. the part in the bracket is a regex group.

#regex groups

a = /[A-Z]\d[A-Z] \d[A-Z]\d/.match(‘asdfasdf M2T5X5 asdfasdf G5G6Gs5’)

This will return M2T5X5

match only returns the first postal code. to find multiple go to ruby docs and look for it.

.gsub – to replace words in strings.

Enumerable – look up ruby docs.

a= [1,2,3,4,5,6,7]

p aselect {|x| x>5}

puts detect

p detect {|x| x>5}

p a.count

p a.count {|x| x> 5}

p a.count do |x| x > 5 end

p a.map {|x| x\*x}.count {|x| x > 10}

.each we use when we want to see the side effects.

it doesn’t change anything in the original array or hash though

its important to remember that it does return something though.

.reduce (aka inject) (aka fold)

a = [1,2,3,3,4,5,6,7,8]

p a.reduce { |memo,x| memo + x } 🡪 returns 36. adds all numbers

first time it runs memo is 1 (index 0). then it adds x to itself. and this memo + x becomes the new memo and the loop continues over the array.

b = [1,6,1,8,9,3,5,8,3]

p b.reduce {|memo,x|

if memo > x

memo

else

x

end

} --- instead of the if else can also do memo > x ? memo : x

c = [{name: “Canada”

cities: [{name: “Toronto”, pop: 1000 }, {name: “Ottawa”, pop: 10}]],

{name: “US”

cities: [{name: “Miami”, pop: 8000 }, {name: “Orlando”, pop: 100}]},

{name: “Germany”

cities: [{name: “Munich”, pop: 6000 }, {name: “Moers”, pop: 500}]}]

result = c.map {|country| country[:cities] }

.flatten

.map

return names of cities of countries with total pop > 4000

result = c.map {|country| country[:total\_population] = country[:cities].map{|city| city[:pop] }.reduce {|m,x| m + x }

country

).select {|country| country[:total\_population]

Blocks