COEN 164 LAB 1

TA: Emma Allegrucci (eallegrucci@scu.edu)
Office Hours: Monday 1:30-2:00pm

A. Basic practice

check out the following code in irb:

```
puts 11.even?
puts 11.odd?
puts 11.class
puts 123456789012345678901234567890.class
puts 11.next
puts 11.succ


12.9.ceil
(-12.9).ceil
-12.9.abs
12.9.floor
12.9.to_i
12.9.to_int
10 ** 2
12.9.round
12.4.round
3.14159.round(2)
4.14159.round(4)


a = 10

a.times { |x|  puts x}
a.times { |x|  print x}

use a.times to calculate  1+2+3+4+5+6+7+8+9+10

a.upto(20) {|x| puts x}
a.upto(1) {|x| puts x}



b = 10..20

b.first
b.last


b.each { |x| puts x}
b.each do |x|
   puts x
end
```

puts ‘it’s a wonderful year’ # how to change this to print: it’s a wonderful year

puts “it’s a wonderful year”

puts %q/it’s a wonderful year/

puts %q/i spent #{a} years to get this degree/
puts %Q/i spent #{a} years to get this degree/
puts %/i spent #{a} years to get this degree/

‘i am ‘ + a.to_s + ‘ years old’

“i am#{a}years old”
“i am ””#{a}”” years old”

“i am” << a.to_s << “ years old”

“cat” <=> “car”

“dog” <=> “fog”

print each char in“abcdefghijklmnopqrstuvwxyz” in a separate line

["apple", “banana”, ”orange”].include?("cherry")

[“Hello”, “from”, “the”, “other”, “side”].join
[“Hello”, “from”, “the”, “other”, “side”].join(“ “)
[“Hello”, “from”, “the”, “other”, “side”].join(“-“)

str = “capital”

str.upcase
str.capitalize
str.capitalize!
str.upcase!

aBcDeFg.swapcase
aBcDeFg.swapcase!

```
arr = %w{d b e f z h a l a b e a z m}
arr.shuffle
arr
arr.shuffle!
arr
arr.slice(4)
arr
arr.slice!(4)
arr
arr.sort
arr
arr.sort!
arr
arr.uniq
arr
arr.uniq!
arr
arr.reverse
arr
arr.reverse!
arr
```

check out the document for Array#split
and give examples of split

check out the document for Array#select
and give examples of select

```
puts “love”.reverse

puts “love”.response_to?(:reverse)

mysymbol = :love

puts mysymbol.reverse


puts mysymbol.response_to?(:reverse)

puts [:a, :b, :c].include?(:a)

["apple", “banana”, ”orange”].include?("cherry")

[“Hello”, “from”, “the”, “other”, “side”].join
[“Hello”, “from”, “the”, “other”, “side”].join(“ “)
[“Hello”, “from”, “the”, “other”, “side”].join(“-“)
```

```
snowy_owl = { "type" => "Bird", "diet" => "Carnivore", "life_span" => "12 years" }
puts snowy_owl[“type"]
snowy_owl["weight"] = “0.5 ounces"
puts snowy_owl
puts snowy_owl.keys
puts snowy_owl.values


x, y = 1, 2
x, y = [1, 2]

a, b = (x, y = 1, 2)
y, x = x, y


def test
  return 1, 2
end
x, y = test

first, second = 1

name = “yuan wang"
first_name, last_name = name.split

puts first_name
puts last_name

n1 = 1
n2 = 2

n1, n2 = n2, n1+ n2

puts n1, n2

n1 = 1
n2 = 2

n1 = n2
n2 = n1+n2

puts n1
puts n2


x, y, z = 1, *[2,3]

x,*y = 1, 2, 3
x,*y = 1, 2
x,*y = 1
x, y, *z = 1, *[2,3,4]
x,(y,z) = a, b
```

```
x,y,z = 1,[2,3]
x,(y,z) = 1,[2,3]
a,b,c,d = [1,[2,[3,4]]]
a,(b,(c,d)) = [1,[2,[3,4]]]


def say(what, *people)
  people.each{|person| puts "#{person}: #{what}"}
end

say "Hello!", "Alice", "Bob", “Carl"

people = ["Rudy", "Sarah", "Thomas"]
say "Howdy!", *people



def arguments_and_opts(*args, opts)
  puts "arguments: #{args} options: #{opts}"
end

arguments_and_opts 1,2,3, :a=>5


def print_pair(a,b,*)
  puts "#{a} and #{b}"
end

print_pair 1,2,3,:cake,7
# 1 and 2


def add(a,b)
  a + b
end

pair = [3,7]
add *pair

a = *(1..3)
a = *[1,2,3]
a = [*[1,2]]
```

B. Write code:

1. write a method that duplicate a string n times , for example

str_dup(5, “click”)

will ouput: click click click click click

2. write a method that calculate sum of squares from 1 to n

for example:

sum_sq(7)

will output the result of : $1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2 + 7^2$

3. For the following hash:

```
h = {  1=>["a", "b"],
       2=>["c"],
       3=>["a", [“d", “f”], "g"],
       4=>["q"]
     }
```

write code to print all the individual character of the hash

4. a = [1, [2,3,4], 5, ,6, [7,8], “love"]

output all element in a, the output should be:

1-2-3-4-5-6-7-8-love-

5. given array of array, for example, [[:key1, value1], [:key2, value2]], use "inject" method to convert this data structure into hash, that is {:key1=>value1, :key2=>value2}