# README

April 23, 2016

## Contents



## 1 Problem 1: Multiples of 3 and 5

If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3,5,6 and 9. The sum of these multiples is 23.

Find the sum of all the multiples of 3 or 5 below 1000.

`./1.rb`

```
multiples = (1..999).select{|i| i%3 == 0 || i%5 == 0}

p multiples.inject(0) {|sum, p| sum+p}
```

## 2 Problem 2: Even Fibonacci Numbers

Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 2, the first 10 terms will be:

```
1,2,3,5,8,13,21,34,55,89, ...
```

By considering the terms in the Fibonacci sequence whose values do not exceed four million, find the sum of the even-valued terms.

./2.rb

```
def fib(n)
  # creates a Fibonacci sequence up to last number <= 4million
  # n = 7

  # [1,2]
  # [1,2,3]
  # [1,2,3,5]
  # [1,2,3,5,8]
  # [1,2,3,5,8,13]
  # [1,2,3,5,8,13,21]

  fib = [1,2]

  while fib.last <= n
    fib.push (fib.last + fib[(fib.count-2)])
  end

  evens = fib.select{|i| i%2==0}

  p evens.inject(0){|sum,x| sum+x}

end

fib(4000000)
```

## 3 Problem 3: Largest Prime Factor

The prime factors of 13195 are 5, 7, 13 and 29.

What is the largest prime factor of the number 600851475143?

./3.rb

```
require 'prime'

def get_factors(n)

  prime_array = []
  p = 2

  if n < 2
    return p
  end

  while p < n
    if n%p == 0 && Prime.prime?(p)
      prime_array.push(p)
      p prime_array
    end
    p +=1
  end

  return prime_array
end

p get_factors(600851475143)
```

## 4 Problem 4: Largest Palindrome Product

A palindromic number reads the same both ways. The largest palindrome made from the product of two 2-digit numbers is $9009 = 91 \times 99$.

Find the largest palindrome made from the product of two 3-digit numbers.

```
./4.rb
```

```
def is_palindrome?(n)
  string = n.to_s
  mid = string.length/2

  a = string[0...mid]

  if string.length.even?
    b = string[mid..-1]
```

```
  else
    b = string[mid+1..-1]
  end

  a == b.reverse
end

def get_factors_max(high,low)

  prods = {}
  pals = []

  high.downto(low).each do |i|

    a = i
    b = i

    until b == low-1
      if is_palindrome?(a*a)
        puts "PAL"
        prods["#{a}*(#{a})"] = a*a
        puts prods["#{a}*(#{a})"]
        pals.push a*a
      else
        if is_palindrome?(a*(b-1))
          puts "PAL"
          prods["#{a}*(#{b-1})"] = a*(b-1)
          puts prods["#{a}*(#{b-1})"]
          pals.push a*(b-1)
        end
      end
      b = b-1
    end

  end

  max = pals.max

  return pals
end
```

```
def largest_palindrome

  a = 999.downto(100).to_a
  a2 = a

  high = 999
  highest_possible = 999*999
  low = 100
  lowest_possible = 100*100

  get_factors_max(high,low)

end

p largest_palindrome
p largest_palindrome.max
```

## 5 Problem 5: Smallest Multiple

2520 is the smallest number that can be divided by each of the numbers from 1 to 10 without any remainder. What is the smallest positive number that is *evenly divisible* by all of the numbers from 1 to 20?

```
   ./5.rb
```

```
i = 20

while (i%2 != 0 ||
         i%3 != 0 ||
           i%4 != 0 ||
             i%5 != 0 ||
               i%6 != 0 ||
                 i%7 != 0 ||
                   i%8 != 0 ||
                     i%9 != 0 ||
                       i%10 != 0 ||
                         i%11 != 0 ||
                           i%12 != 0 ||
                             i%13 != 0 ||
                               i%14 != 0 ||
```

```
                                    i%15 != 0 ||
                                      i%16 != 0 ||
                                        i%17 != 0 ||
                                          i%18 != 0 ||
                                            i%19 != 0 ||
                                              i%20 != 0)
    i = i+1
  end

p i
```

## 6 Problem 6: Sum Square Difference

The sum of the squares of the first ten natural numbers is,

```
1^2 + 2^2 + ... + 10^2 = 385
```

The square of the sum of the first ten natural numbers is,

```
(1 + 2 + ... + 10)^2 = 55^2 = 3025
```

Hence the difference between the sum of the squares of the first ten natural numbers and the square of the sum is 3025 - 385 = 2640.

Find the difference between the sum of the squares of the first one hundred natural numbers and the square of the sum.

`./6.rb`

```
range = (1..100)
squares = range.map { |i| i*i }
sum_squares = squares.inject(0) { |sum, i| sum + i }

sum = range.inject(0) { |sum, i| sum + i }

p sum**2 - sum_squares
```

## 7 Problem 7: 10001st prime

By listing the first six prime numbers: 2,3,5,7,11 and 13, we can see that the 6th prime is 13.

What is the 10001st prime number?

`./7.rb`

```
require 'prime'


p ((1..105000).select {|p| Prime.prime?(p)})[10000]
```

## 8 Problem 8: Largest Product in a Series

The four adjacent digits in the 1000-digit number that have the greatest product are 9*9*8*9=5832.

```
73167176531330624919225119674426574742355349194934
96983520312774506326239578318016984801869478851843
85861560789112949495459501737958331952853208805511
12540698747158523863050715693290963295227443043557
66896648950445244523161731856403098711121722383113
62229893423380308135336276614282806444486645238749
30358907296290491560440772390713810515859307960866
70172427121883998797908792274921901699720888093776
65727333001053367881220235421809751254540594752243
52584907711670556013604839586446706324415722155397
53697817977846174064955149290862569321978468622482
83972241375657056057490261407972968652414535100474
82166370484403199890008895243450658541227588666881
16427171479924442928230863465674813919123162824586
17866458359124566529476545682848912883142607690042
24219022671055626321111109370544217506941658960408
07198403850962455444362981230987879927244284909188
84580156166097919133875499200524063689912560717606
05886116467109405077541002256983155200055935729725
71636269561882670428252483600823257530420752963450
```

Find the thirteen adjacent digits in the 1000-digit number that have the greatest product. What is the value of this product?

```
./8.rb
```

```
num = "731671765313306249192251196744265747423553491949349698352031277450632623957831801
```

```
def get_products(a)
  a.inject(1) {|prod,x| prod*x}
end
```

```
def build_original(n)
  original = []

  n.split("").each do |i|
    original.push i.to_i
  end

  return original
end

def get_thirteen(n)

  original = build_original(n)
  set,sets = [],[]

  new = original
  count = new.size

  while count >= 13
    sets.push get_products(new[0...13])
    new.shift
    count = new.size
  end

  return sets.max
end

p get_thirteen(num)
```

## 9 Problem 9: Special Pythagorean Triplet

A Pythagorean triplet is a set of three natural numbers, $a < b < c$, for which,

```
a**2 + b**2 == c**2
```

For example,

```
3**2 + 4**2 == 9 + 16 == 25 == 5**2
```

There exists exactly one Pythagorean triplet for which $a + b + c = 1000$ Find the product $abc$.

```
./9.rb
```

```
def triplet?(a,b,c)
  a**2 + b**2 == c**2
end

def is_the_one?(a,b,c)
  a+b+c == 1000
end

def find_triplets

  range = (2..999)
  sets = []

  range.each do |i|
    # i = 1
    count = 1000
    # 1000

    while count > 0
      set = []
      max = range.size

      a = i
      b = count-a
      c = 1000-(b+a)

      if a+b+c==1000 && (a**2 + b**2 == c**2)
        set.push a,b,c

        sets << a*b*c if a*b*c > 0
      end

      count = count-1
    end
  end
```

```
  return sets.uniq.first
end
p find_triplets
```