

README

April 22, 2016

Contents

1	Problem 1: Multiples of 3 and 5	1
2	Problem 2: Even Fibonacci Numbers	2
3	Problem 3: Largest Prime Factor	2
4	Problem 4: Largest Palindrome Product	3
5	Problem 5: Smallest Multiple	5
6	Problem 6: Sum Square Difference	6
7	Problem 7: 10001st prime	6
8	Problem 8: Largest Product in a Series	7
	Project Euler Problems	

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,

$$1^2 + 2^2 + \dots + 10^2 = 385$$

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

$$(1 + 2 + \dots + 10)^2 = 55^2 = 3025$$

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)}).first(10000)
```

8 Problem 8: Largest Product in a Series

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

```
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 = "7316717653133062491922511967442657474235534919493496983520312774506326239578318016984801869478851843858615607891129494954595017379583319528532088055111254069874715852386305071569329096329522744304355766896648950445244523161731856403098711121722383113622298934233803081353362766142828064444866452387493035890729629049156044077239071381051585930796086670172427121883998797908792274921901699720888093776657273330010533678812202354218097512545405947522435258490771167055601360483958644670632441572215539753697817977846174064955149290862569321978468622482839722413756570560574902614079729686524145351004748216637048440319989000889524345065854122758866688116427171479924442928230863465674813919123162824586178664583591245665294765456828489128831426076900422421902267105562632111110937054421750694165896040807198403850962455444362981230987879927244284909188845801561660979191338754992005240636899125607176060588611646710940507754100225698315520005593572972571636269561882670428252483600823257530420752963450"
```

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

```

def get_thirteen(n)
  original = []
  set = []
  sets = []

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

  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)

```