## Harshad Numbers

### Problem 387

A **Harshad or Niven number** is a number that is divisible by the sum of its digits.   
201 is a Harshad number because it is divisible by 3 (the sum of its digits.)   
When we truncate the last digit from 201, we get 20, which is a Harshad number.   
When we truncate the last digit from 20, we get 2, which is also a Harshad number.   
Let's call a Harshad number that, while recursively truncating the last digit, always results in a Harshad number a *right truncatable Harshad number.*

Also:   
201/3=67 which is prime.   
Let's call a Harshad number that, when divided by the sum of its digits, results in a prime a *strong Harshad number*.

Now take the number 2011 which is prime.   
When we truncate the last digit from it we get 201, a strong Harshad number that is also right truncatable.   
Let's call such primes *strong, right truncatable Harshad primes*.

You are given that the sum of the strong, right truncatable Harshad primes less than 10000 is 90619.

Find the sum of the strong, right truncatable Harshad primes less than 1014.

Code: 387

@language = python

import sys

import random

def toBinary(n):

r = []

while (n > 0):

r.append(n % 2)

n = n / 2

return r

def test(a, n):

b = toBinary(n - 1)

d = 1

for i in xrange(len(b) - 1, -1, -1):

x = d

d = (d \* d) % n

if d == 1 and x != 1 and x != n - 1:

return True # Complex

if b[i] == 1:

d = (d \* a) % n

if d != 1:

return True # Complex

return False # Prime

def MillerRabin(n, s = 50):

if n<10:

if n in [2,3,5,7]:

return True

else: return False

for j in xrange(1, s + 1):

a = random.randint(1, n - 1)

if (test(a, n)):

return False # n is complex

return True # n is prime

harsh = [1,2,3,4,5,6,7,8,9]

nHrsh = []

pSrth = []

srth = 0

def digSum(n):

sm = 0

while n:

sm += n%10

n = n/10

return sm

def lrHarsh(n):

global harsh, nHrsh, srth

if not n: return

for num in harsh:

nHrsh.append(10\*num)

for i in xrange(1,10):

test = 10\*num+i

if MillerRabin(test):

if MillerRabin(num/digSum(num)):

srth += test

pSrth.append(test)

if not test % digSum(test):

nHrsh.append(test)

harsh = nHrsh

nHrsh = []

n -= 1

lrHarsh(n)

lrHarsh(13)

print srth

#print harsh

#print pSrth

Time taken:

real 0m13.227s

user 0m13.157s

sys 0m0.032s

## (prime-k) factorial

### Problem 381

For a prime p let S(p) = (![A description...](data:None;base64,)(p-k)!) mod(p) for 1 ![A description...](data:None;base64,) k ![A description...](data:None;base64,) 5.

For example, if p=7,  
(7-1)! + (7-2)! + (7-3)! + (7-4)! + (7-5)! = 6! + 5! + 4! + 3! + 2! = 720+120+24+6+2 = 872.  
As 872 mod(7) = 4, S(7) = 4.

It can be verified that ![A description...](data:None;base64,)S(p) = 480 for 5 ![A description...](data:None;base64,) p ![A description...](data:None;base64,) 100.

Find ![A description...](data:None;base64,)S(p) for 5 ![A description...](data:None;base64,) p ![A description...](data:None;base64,) 108.

Code: 381

@language = pari / gp

default(primelimit,100000000);

S(p) = {

a=Mod(-1,p);

b=a;

for(k=1,4,

b=b+a/(p-k);

a=a/(p-k));

lift(b)

}

c=0;

forprime(x=5,10^8,c=c+S(x));

c

# Note: can reduce the calculation from S(p), as first two terms always sum to zero

Time taken: ~ 10 sec

## Prime generating integers

### Problem 357

Consider the divisors of 30: 1,2,3,5,6,10,15,30.  
It can be seen that for every divisor *d* of 30, *d*+30/*d* is prime.

Find the sum of all positive integers *n* not exceeding 100 000 000  
such that for every divisor *d* of *n*, *d*+*n*/*d* is prime.

Code: 357

@language = pari / gp

default(primelimit,100000000)

mx=100000000;

sm=0;

{

forprime(i=1,mx,

n=i-1;

fl=1;

if(moebius(n)==0,next);

fordiv(n,j,

if(isprime(j+n/j)==0,fl=0;break);

if(j>sqrt(n),break)

);

sm+=fl\*n

)

}

sm

Time taken: ~ 5 sec

## Largest integer divisible by two primes

### Problem 347

The largest integer ![A description...](data:None;base64,) 100 that is only divisible by both the primes 2 and 3 is 96, as 96=32\*3=25\*3. For two *distinct* primes p and q let M(p,q,N) be the largest positive integer ![A description...](data:None;base64,)N only divisible by both p and q and M(p,q,N)=0 if such a positive integer does not exist.

E.g. M(2,3,100)=96.  
M(3,5,100)=75 and not 90 because 90 is divisible by 2 ,3 and 5.  
Also M(2,73,100)=0 because there does not exist a positive integer ![A description...](data:None;base64,) 100 that is divisible by both 2 and 73.

Let S(N) be the sum of all distinct M(p,q,N). S(100)=2262.

Find S(10 000 000).

Code: 347

n=10000000;

sm=0;

{

forprime(i=1,floor(sqrt(n)),

forprime(j=i+1,(floor(n/i)),

pr=i\*j;

k=n-n%pr;

while(1,

fac=factor(k)[,1];

prd=prod(l=1,#fac,fac[l]);

if(prd==pr,sm+=k;break);

k-=pr;

)))

}

Time taken: ~10 sec

## Prime generating integers

### Problem 357

Consider the divisors of 30: 1,2,3,5,6,10,15,30.  
It can be seen that for every divisor *d* of 30, *d*+30/*d* is prime.

Find the sum of all positive integers *n* not exceeding 100 000 000  
such that for every divisor *d* of *n*, *d*+*n*/*d* is prime.

Code: 357

@language = python

from munkres import Munkres

matrix = [[ -negated original matrix ]]

m = Munkres()

indexes = m.compute(matrix)

total = 0

for row, column in indexes:

value = matrix[row][column]

total += value

print '(%d, %d) -> %d' % (row, column, -value)

print 'total cost: %d' % -total

Time taken:

real 0m0.041s

user 0m0.020s

sys 0m0.016s

## Angular Bisector and Tangent

### Problem 296

Given is an integer sided triangle *ABC* with *BC* ![A description...](data:None;base64,) *AC* ![A description...](data:None;base64,) *AB*.  
*k* is the angular bisector of angle *ACB*.  
*m* is the tangent at *C* to the circumscribed circle of *ABC*.  
*n* is a line parallel to *m* through *B*.  
The intersection of *n* and *k* is called *E*.

![A description...](data:None;base64,)

How many triangles *ABC* with a perimeter not exceeding 100 000 exist such that *BE* has integral length?

Code: 296

@language = pari / gp

lim=100000;

cnt=0;

{

for(a=1,1+lim/3,

b=2\*a;

prms=mattranspose(factor(a)[,1]);

mods=prms;

for(i=1,length(prms),

mods[i]=Mod(b,prms[i]));

while(b<=min(a\*a,a+(lim-a)/2),

c=b-a;

fct=(a+c)/gcd(a,a+c);

maxc=min(a+c-1,lim-a-c);

rng=maxc-c+lift(Mod(c,fct));

smcnt=if(rng<0,0,if(Mod(c,fct)==0,1,0)+floor(rng/fct)

);

cnt+=smcnt;

elm=vecmin(prms-lift(mods));

for(i=1,length(prms),

mods[i]+=elm

);

b+=elm;

));

}

Time taken: ~10 mins !!

## Bitwise-OR operations on random integers

### Problem 323

Let *y*0, *y*1, *y*2,... be a sequence of random unsigned 32 bit integers  
(i.e. 0  *yi*  232, every value equally likely).

For the sequence *xi* the following recursion is given:

* *x*0 = 0 and
* *xi* = *xi*-*1* **|** *yi*-*1*, for *i*  0. ( **|** is the bitwise-OR operator)

It can be seen that eventually there will be an index N such that *xi* = 232 -1 (a bit-pattern of all ones) for all *i*  N.

Find the expected value of N.   
Give your answer rounded to 10 digits after the decimal point.

Code: 296

@language = python

from operator import mul

nCk = lambda n,k: int(round(

reduce(mul, (float(n-i)/(i+1) for i in range(k)), 1)

))

ans = [2]

for i in xrange(2,33):

tmp = 2.0/2\*\*i

for j in xrange(len(ans)):

tmp += nCk(i,j+1)\*(1.0/2\*\*i)\*(1+ans[j])

new = (1/(1-1.0/2\*\*i))\*tmp

print i, new

ans.append(new)

Time taken: ~5 s