Problems

Amicable numbers

Problem 21: Let $d(n)$ be defined as the sum of proper divisors of n (numbers less than n which divide evenly into n). If $d(a) = b$ and $d(b) = a$, where $a \neq b$, then a and b are an amicable pair and each of a and b are called amicable numbers.

For example, the proper divisors of 220 are $1, 2, 4, 5, 10, 11, 20, 22, 44, 55$ and 110; therefore $d(220) = 284$. The proper divisors of 284 are $1, 2, 4, 71$ and 142; so $d(284) = 220$.

Evaluate the sum of all the amicable numbers under 10000.

```python
from timing import timing_function
from arithmetic import divisors, triangular
from typing import List

def sumOfProperDivisors(a : int) -> int:
    return sum(divisors(a)) - a

def isAmicable(a : int) -> bool:
    b = sumOfProperDivisors(a)
    if b == a:
        return False
    elif a == sumOfProperDivisors(b):
        return True
    else:
        return False

def euler_21():
    accum = 0
    for n in range(2, 10000):
        if isAmicable(n):
            accum += n
    return accum


def main():
    print(timing_function(euler_21))

main()
```

```
>>> 31626
Time it took to run the function: 0.1475389003753662 seconds
```

Highly divisible triangular number

Problem 12: The sequence of triangle numbers is generated by adding the natural numbers. So the 7th triangle number would be $1 + 2 + 3 + 4 + 5 + 6 + 7 = 28$. The first ten terms would be:

$$1, 3, 6, 10, 15, 21, 28, 36, 45, 55, \ldots$$

Let us list the factors of the first seven triangle numbers:

1:	1					
3:	1,	3				
6:	1,	2,	3,	6		
10:	1,	2,	5,	10		
15:	1,	3,	5,	15		
21:	1,	3,	7,	21		
28:	1,	2,	4,	7,	14,	28

We can see that 28 is the first triangle number to have over five divisors.

What is the value of the first triangle number to have over five hundred divisors?

```python
from timing import timing_function
from arithmetic import divisors, triangular


def euler_12():
    n = 1
    while len(divisors(triangular(n))) <= 500:
        n += 1
    return triangular(n)


def main():
    print(timing_function(euler_12))

main()

>>>
76576500
Time it took to run the function: 12.990790843963623 seconds
```

Large Sum

PROBLEM 13: Work out the first ten digits of the sum of the following one-hundred 50-digit numbers.

37107287533902102798797998220837590246510135740250
46376937677490009712648124896970078050417018260538
74324986199524741059474233309513058123726617309629
91942213363574161572522430563301811072406154908250
23067588207539346171171980310421047513778063246676
89261670696623633820136378418383684178734361726757
28112879812849979408065481931592621691275889832738
44274228917432520321923589422876796487670272189318
47451445736001306439091167216856844588711603153276
70386486105843025439939619828917593665686757934951
62176457141856560629502157223196586755079324193331
64906352462741904929101432445813822663347944758178
92575867718337217661963751590579239728245598838407
58203565325359399008402633568948830189458628227828
80181199384826282014278194139940567587151170094390
35398664372827112653829987240784473053190104293586
86515506006295864861532075273371959191420517255829
71693888707715466499115593487603532921714970056938
54370070576826684624621495650076471787294438377604
53282654108756828443191190634694037855217779295145
36123272525000296071075082563815656710885258350721
45876576172410976447339110607218265236877223636045
17423706905851860660448207621209813287860733969412
81142660418086830619328460811191061556940512689692
51934325451728388641918047049293215058642563049483
62467221648435076201727918039944693004732956340691
15732444386908125794514089057706229429197107928209
55037687525678773091862540744969844508330393682126
18336384825330154686196124348767681297534375946515
80386287592878490201521685554828717201219257766954
78182833757993103614740356856449095527097864797581
16726320100436897842553539920931837441497806860984
48403098129077791799088218795327364475675590848030
87086987551392711854517078544161852424320693150332
59959406895756536782107074926966537676326235447210
69793950679652694742597709739166693763042633987085
41052684708299085211399427365734116182760315001271
65378607361501080857009149939512557028198746004375
35829035317434717326932123578154982629742552737307
94953759765105305946966067683156574377167401875275
88902802571733229619176668713819931811048770190271
25267680276078003013678680992525463401061632866526
36270218540497705585629946580636237993140746255962
24074486908231174977792365466257246923322810917141
91430288197103288597806669760892938638285025333403
34413065578016127815921815005561868836468420090470
23053081172816430487623791969842487255036638784583
11487696932154902810424020138335124462181441773470
63783299490636259666498587618221225225512486764533
67720186971698544312419572409913959008952310058822
95548255300263520781532296796249481641953868218774
76085327132285723110424803456124867697064507995236
37774242535411291684276865538926205024910326572967
23701913275725675285653248258265463092207058596522
29798860272258331913126375147341994889534765745501
18495701454879288984856827726077713721403798879715
38298203783031473527721580348144513491373226651381
34829543829199918180278916522431027392251122869539
40957953066405232632538044100059654939159879593635
29746152185502371307642255121183693803580388584903

```
41698116222072977186158236678424689157993532961922
62467957194401269043877107275048102390895523597457
23189706772547915061505504953922979530901129967519
86188088225875314529584099251203829009407770775672
11306739708304724483816533873502340845647058077308
82959174767140363198008187129011875491310547126581
97623331044818386269515456334926366572897563400500
42846280183517070527831839425882145521227251250327
55121603546981200581762165212827652751691296897789
32238195734329339946437501907836945765883352399886
75506164965184775180738168837861091527357929701337
62177842752192623401942399639168044983993173312731
32924185707147349566916674687634660915035914677504
99518671430235219628894890102423325116913619626622
73267460800591547471830798392868535206946944540724
76841822524674417161514036427982273348055556214818
97142617910342598647204516893989422179826088076852
87783646182799346313767754307809363333018982642090
10848802521674670883215120185883543223812876952786
71329612474782464538636993009049310363619763878039
62184073572399794223406235393808339651327408011116
66627891981488087797941876876144230030984490851411
60661826293682836764744779239180335110989069790714
85786944089552990653640447425576083659976645795096
66024396409905389607120198219976047599490197230297
64913982680032973156037120041377903785566085089252
16730939319872750275468906903707539413042652315011
94809377245048795150954100921645863754710598436791
78639167021187492431995700641917969777599028300699
15368713711936614952811305876380278410754449733078
40789923115535562561142322423255033685442488917353
44889911501440648020369068063960672322193204149535
41503128880339536053299340368006977710650566631954
81234880673210146739058568557934581403627822703280
82616570773948327592232845941706525094512325230608
22918802058777319719839450180888072429661980811197
77158542502016545090413245809786882778948721859617
72107838435069186155435662884062257473692284509516
20849603980134001723930671666823555245252804609722
53503534226472524250874054075591789781264330331690
```

```
from timing import timing_function
from arithmetic import divisors, triangular

def euler_13():
    accum = 0
    with open("euler_13.txt", 'r') as f:
        line = f.readline().rstrip()
        while line:
            accum +=int(line)
            line = f.readline().rstrip()
        return str(accum)[0:10]

def main():
    print(timing_function(euler_13))

main()

>>>
5537376230
Time it took to run the function: 0.04814600944519043 seconds
```

Longest Collatz sequence

Problem 14: The following iterative sequence is defined for the set of positive integers:

$$n \to n/2 \quad (n \text{is even})$$
$$n \to 3n + 1 (n \text{is odd})$$

Using the rule above and starting with 13, we generate the following sequence:

$$13 \to 40 \to 20 \to 10 \to 5 \to 16 \to 8 \to 4 \to 2 \to 1$$

It can be seen that this sequence (starting at 13 and finishing at 1) contains 10 terms. Although it has not been proved yet (Collatz Problem), it is thought that all starting numbers finish at 1.

Which starting number, under one million, produces the longest chain?

NOTE: Once the chain starts the terms are allowed to go above one million.

```
import time
from arithmetic import divisors, triangular

def chainLength(n):
    count  = 1
    while n > 1:
        if n % 2 == 0:
            n = n // 2
        else:
            n = 3*n + 1
        count += 1
    return count

def euler_14(n = 1000000):
    aux = 0
    for n in range(n+1):
        a = chainLength(n)
        if a > aux:
            aux = a
            value = n
    return value

def main():
        t1 = time.time()
        print(euler_14())
        t2 = time.time()
        print("Time it took to run the function: " + str((t2 - t1)) + " seconds")
main()

>>>
837799
Time it took to run the function: 22.4543559551239 seconds
```

Lattice paths

PROBLEM 15: Starting in the top left corner of a 2×2 grid, and only being able to move to the right and down, there are exactly 6 routes to the bottom right corner.

How many such routes are there through a 20×20 grid?

```
from timing import timing_function
from arithmetic import binomial

def euler_15():
    return binomial(20 + 20, 20)

def main():
    print(timing_function(euler_15))

main()

137846528820
Time it took to run the function: 0.07077383995056152 seconds
```

Power digit sum

PROBLEM 16: $2^{15} = 32768$ and the sum of its digits is $3 + 2 + 7 + 6 + 8 = 26$.

What is the sum of the digits of the number 2^{1000}?

```
from timing import timing_function
from arithmetic import sumOfDigits

def euler_16():
    return sumOfDigits(2**1000)
```

```
def main():
    print(timing_function(euler_16))

main()

>>>
1366
Time it took to run the function: 0.06404399871826172 seconds
```

Quadratic primes

Problem 27: Euler discovered the remarkable quadratic formula:

$$n^2 + n + 41$$

If all the numbers from 1 to 1000 (one thousand) inclusive were written out in words, how many letters would be used? It turns out that the formula will produce 40 primes for the consecutive integer values $0 \le n \le 39$. However, when $n = 40, 40^2 + 40 + 41 = 40(40+1) + 41$ is divisible by 41, and certainly when $n = 41, 41^2 + 41 + 41$ is clearly divisible by 41.

The incredible formula $n^2 - 79n + 1601$ was discovered, which produces 80 primes for the consecutive values $0 \le n \le 79$. The product of the coefficients, -79 and 1601, is -126479.

Considering quadratics of the form:

$$n^2 + an + b, \text{ where } |a| < 1000 \text{ and } |b| \le 1000$$

where $|n|$ is the modulus/absolute value of n e.g. $|11| = 11$ and $|-4| = 4$

Find the product of the coefficients, a and b, for the quadratic expression that produces the maximum number of primes for consecutive values of n, starting with $n = 0$.

```
from timing import timing_function
from arithmetic import primes, isPrime

primes = primes(1000)
while primes[0] <= 40:
    primes.pop(0)

primes.extend([-p for p in primes])

def quadratic(a : int , b : int , n : int):
    return n**2 + a*n + b

def consecutive(a : int , b : int):
    n = 0
    while isPrime(quadratic(a, b, n)):
        n +=1
    return n

def euler_27():
    maxConsecutive = -1
    for a in range(-999, 1000):
        for b in primes:
            aux = consecutive(a, b)
            if aux > maxConsecutive:
                maxA, maxB, maxConsecutive = a, b, aux
    return maxA * maxB

>>> -59231
Time it took to run the function: 3.9455649852752686 seconds
```

Recipe 18.10: Computing Prime Numbers

Maximum path sum I

PROBLEM 18: By starting at the top of the triangle below and moving to adjacent numbers on the row below, the maximum total from top to bottom is 23.

```
      3
    7   4
  2   4   6
8   5   9   3
```

That is, $3 + 7 + 4 + 9 = 23$.

Find the maximum total from top to bottom of the triangle below:

```
                            75
                          95 64
                        17 47 82
                      18 35 87 10
                    20 04 82 47 65
                  19 01 23 75 03 34
                88 02 77 73 07 63 67
              99 65 04 28 06 16 70 92
            41 41 26 56 83 40 80 70 33
          41 48 72 33 47 32 37 16 94 29
        53 71 44 65 25 43 91 52 97 51 14
      70 11 33 28 77 73 17 78 39 68 17 57
    91 71 52 38 17 14 91 43 58 50 27 29 48
  63 66 04 68 89 53 67 30 73 16 69 87 40 31
04 62 98 27 23 09 70 98 73 93 38 53 60 04 23
```

NOTE: As there are only 16384 routes, it is possible to solve this problem by trying every route. However, Problem 67, is the same challenge with a triangle containing one-hundred rows; it cannot be solved by brute force, and requires a clever method!

```
combine us []            = us
combine us [_]           = us
combine us (x : y : xs) = (head us + maximum [x, y]) : combine (tail us) (y : xs)

main18 = do
    scr <- readFile "problem18.txt"
    let qs = lines scr
    let qss = map words qs
    let ts = map (\ls -> map strToInt ls) qss
    let sol = foldr combine [] ts
    putStr (show (head sol))
    putStr "\n"
```

```
*Main> main18
1074
```

```
from timing import timing_function
from lists import product

def max_product(ls, n, start = 1):
    """
    Returns the greatest product  of n consecutive numbers in the list ls

    When the length of ls is less than n, returns a 'start' value (default: 1) value.
    This function is intended specifically for use with positive numeric values and may
    reject non-numeric types.
    """
    while (len(ls) >= n):
```

```
        start = max(product(ls[0:n]), start)
        return max_product(ls[1:], n, start)
    return start

def euler_08():
    ls = list()
    with open("euler_08.txt") as f:
        line = f.readline().rstrip()
        while line:
            ls += list(map(int, list(line)))
            line = f.readline().rstrip()
        return max_product(ls, 13)

def main():
    print(timing_function(euler_08))

main()
```

```
>>>
23514624000
Time it took to run the function: 0.06601810455322266 seconds
```

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.

```
from timing import timing_function

def euler_09():
    for a in range(1, 1001):
        for b in range(a, 1001):
            c = 1000 - a - b
            if b < c:
                if a**2 + b**2 == c**2:
                    return a*b*c
            else:
                break

def main():
    print(timing_function(euler_09))

main()
```

```
>>>
31875000
Time it took to run the function: 0.12091183662414551 seconds
```

Summation of primes

The sum of the primes below 10 is $2 + 3 + 5 + 7 = 17$. Find the sum of all the primes below two million.

```
from timing import timing_function
from arithmetic import isPrime

def euler_10():
    accum = 2
    for n in range(3, 2000000, 2):
        if isPrime(n):
            accum += n
```

```python
    return accum

def main():
    print(timing_function(euler_10))

main()
```

```
>>>
142913828922
Time it took to run the function: 13.390980243682861 seconds
```