Питон, теория чисел

Хашин С.И.
http://math.ivanovo.ac.ru/dalgebra/Khashin/index.html

Ивановский университет

.

Питон, теория чисел

.

Иваново-2023

План

Прямая проверка простоты

```
def is_prime(n):    # будет ли число n простым
    if n<2: return False
    for k in range(2, n):
        if n%k==0: return False
    return True

print(is_prime(101))
print(is_prime(103))
print(is_prime(1000003))
print(is_prime(1000000007))
```

Прямая проверка простоты

```
def is_prime(n):    # будет ли число n простым
    if n<2: return False
    for k in range(2, n):
        if n%k==0: return False
        if k*k>n: return True
    return True

print(is_prime(1000003))
print(is_prime(1000000007))
```

Эратосфен

```
def primes(n):    # список простых <= n
    res = [2,3]
    for k in range(5, n, 2):
        k_prime = True
        for p in res:
            if k%p==0: k_prime=False; break
            if p*p>k: break
        if k_prime: res.append(k)
    return res

print(primes(30))
> [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]
pp = primes(10**6)
print(len(pp), pp[-1])
> 78498 999983
```

Ферма

Малая теорема Ферма. Пусть p — простое число и a не делится на p. Тогда

$$a^{p-1} mod p = 1.$$

```
def Ferma2(n):
    return pow(2,n-1,n)==1

print(Ferma2(11))
print(Ferma2(101))
print(Ferma2(1001))
print(Ferma2(1000003))
print(Ferma2(1000000007))
```

Ферма

Ошибка в функции Ferma2:

```
for k in range(30,1000,3):
    if Ferma2(k): print(k)
```

```
 561
 645
```

```
def Ferma(n, L):
  for a in L:
    if pow(a,n-1,n) != 1: return False
  return True

 print(Ferma(341,[2,3]))
```

hnumber_theory

```
Ferma(n, L) Fermat primality test of n by bases in list L
Miller_Rabin(n, L) M-R primality test of n by bases in L
Frobenius(n) Frobenius primality test

import hnumber_theory as nth
print(nth.Ferma(561, [2]))
print(nth.Ferma(561, [2,3]))
print(nth.Ferma(561, [2,3,5]))
```

hnumber_theory

```
n = 2845963*11383849
print('n=', n)
print(nth.Ferma(n, [2]))
print(nth.Ferma(n, [2,3]))
print(nth.Ferma(n, [2,3,5]))
print(nth.Ferma(n, [2,3,5,7]))
print(nth.Ferma(n, [2,3,5,7,11]))
print(nth.Ferma(n, [2,3,5,7,11,13]))
print(nth.Ferma(n, [2,3,5,7,11,13,17]))
```

```
> n= 32398013051587
> True True True True True True
> False
```

Задачи

- Найти простые вида $10 * *s + k$.
- Найти псевдопростые для L=[2] вида 3*n
- Найти псевдопростые для L=[2,3] вида 5*n
- Найдите натуральные числа p такие, что 2**(p-1) сравнимо с 1 по модулю p*p.
- Найдите простые числа p от 1000_000 до 1001_000 такие, что p+2 тоже простое.
- Найдите простые числа p от 1000_000 до 1001_000 такие, что 2p+1 тоже простое.

hnumber_theory2

В модуле hnumber_theory2.py есть функция разложения на множители Factor:

```
import hnumber_theory2 as nth
N = 10**6
for k in range(30):
    print(k, nth.Factor(N+k))

for k in range(20):
 n = (10**k-1)//3 -2
 print(n, nth.Factor(n))
```

Задачи

- Разложить на множители числа вида $11, 111, 1111, \ldots$.
- Разложить на множители числа вида $31, 331, 3331, \ldots$.
- Разложить на множители числа вида $2^k + 1$ при $k = 1, 2, \ldots$.
- Разложить на множители числа вида $2^{2^k} + 1$ при $k = 1, 2, \ldots$.
- Разложить на множители числа вида $10^k + 1$ при $k = 1, 2, \ldots$.
- Разложить на множители числа из файла pq0.txt.