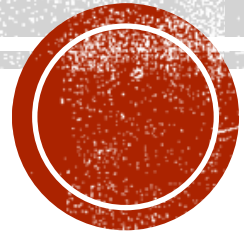
ОСОБЕННОСТИ РЕШЕНИЯ 25 ЗАДАНИЯ



Анищенко Юлия Михайловна

Учитель информатики ГБОУ лицей №64

Спецификация

Задание 25 (высокий уровень)

Время выполнения: 20 минут

Тема: Обработка целых чисел. Проверка делимости

Что проверяется:

Умение создавать собственные программы (10–20 строк) для обработки целочисленной информации.

Демонстрационный вариант

Задание 25 (высокий уровень)

Назовём маской числа последовательность цифр, в которой также могут встречаться следующие символы:

- символ «?» означает ровно одну произвольную цифру;
- символ «*» означает любую последовательность цифр произвольной длины; в том числе «*» может задавать и пустую последовательность.

Например, маске 123*4?5 соответствуют числа 123405 и 12300425.

Среди натуральных чисел, не превышающих 10^{10}, найдите все числа, соответствующие маске 1?2139*4, делящиеся на 2023 без остатка.

В ответе запишите в первом столбце таблицы все найденные числа в порядке возрастания, а во втором столбце — соответствующие им частные от деления на 2023.

Решение (программа, Python? 1 способ)

Для решения используем **f-строки.** Форматирование, которое появилось в **Python 3.6** (PEP 498). Этот способ похож на форматирование с помощью метода **format()**, но гибче, читабельней и быстрей.

```
for i in '0123456789':
    s = int(f'1{i}21394')
    if s%2023==0:
        print(s,s//2023)
```

f-строки берут значения переменных, которые есть в текущей области видимости, и подставляют их в строку. В самой строке нужно лишь указать имя этой переменной в фигурных скобках.

Решение (программа, Python? 1 способ)

```
for i in '0123456789':
    for j in '0123456789':
        for k in '0123456789':
            s = int(f'1{i}2139{j}{k}4')
            if s % 2023 == 0:
                print(s, s // 2023)

for i in '0123456789':
    for j in '0123456789':
        for k in '0123456789':
            for h in '0123456789':
                s = int(f'1{i}2139{j}{k}{h}4')
                if s % 2023 == 0:
                    print(s, s // 2023)
```

Решение (программа, Python, 2 способ)

fnmatch.fnmatch(*filename, pattern*)

Параметры:

- **filename** - строка которую проверяем,
- **pattern** - строка шаблона.

Возвращаемое значение: bool.

Функция fnmatch() модуля fnmatch проверяет, соответствует ли строка, которую проверяем шаблонной строке, возвращая True или False.

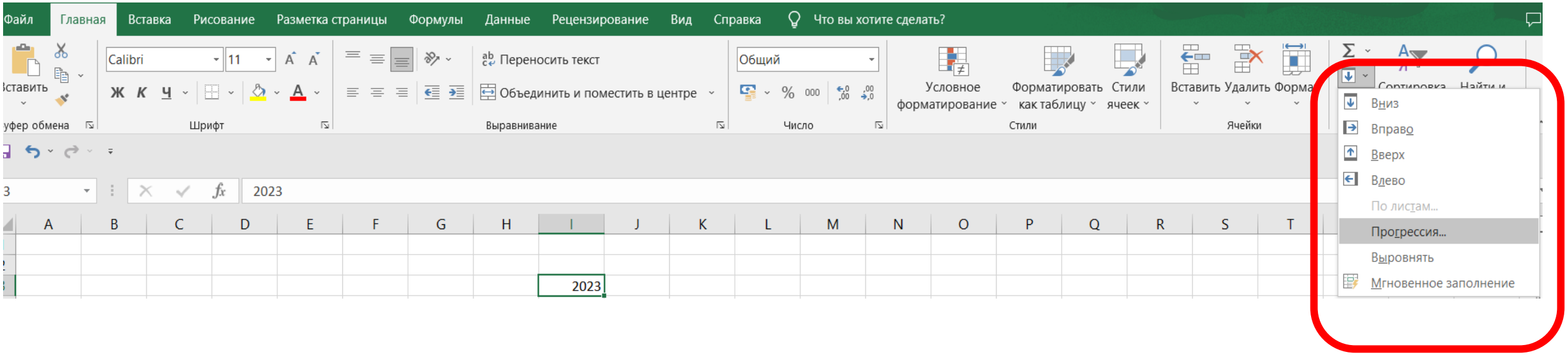
```
from fnmatch import *

for x in range(0, 10**10, 2023):
    if fnmatch(str(x), '1?2139*4'):
        print(x, x//2023)
```

Решение (Таблицы, Excel, 3 способ)

Сгенерируем последовательность из всех чисел, которые нам могут подойти (с помощью прогрессии). Так как числа должны делиться на 2023, то мы будем проверять не все 10^{10} чисел, а только кратные 2023: 2023, 4046, 6069 …

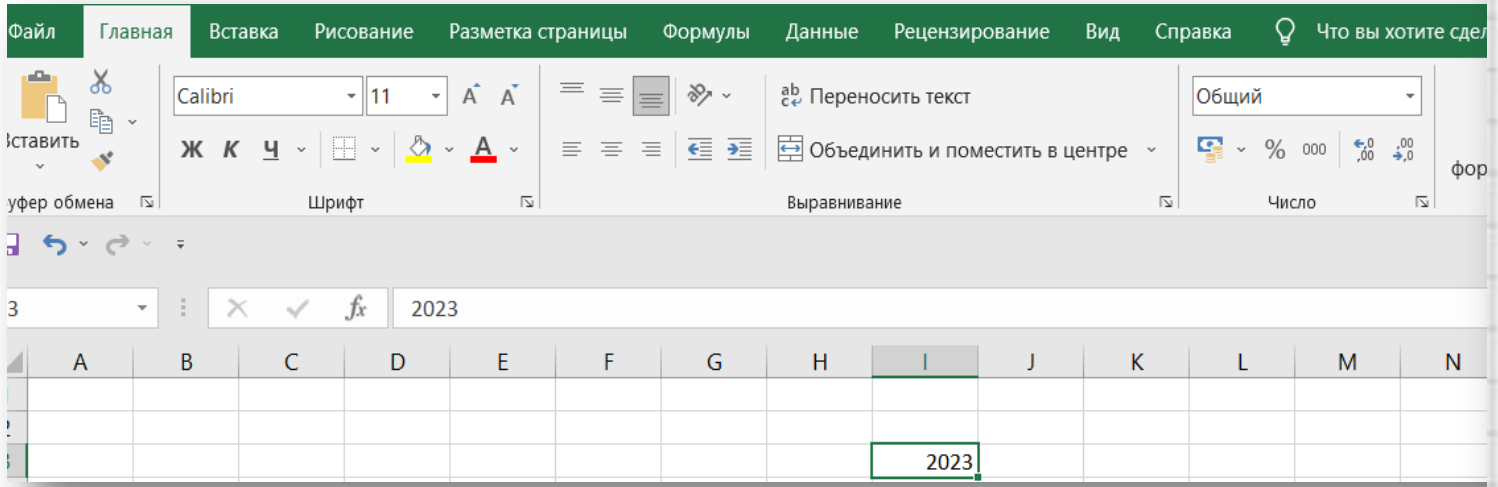
В ячейку записываем число 2023 и нажимаем «прогрессия…»

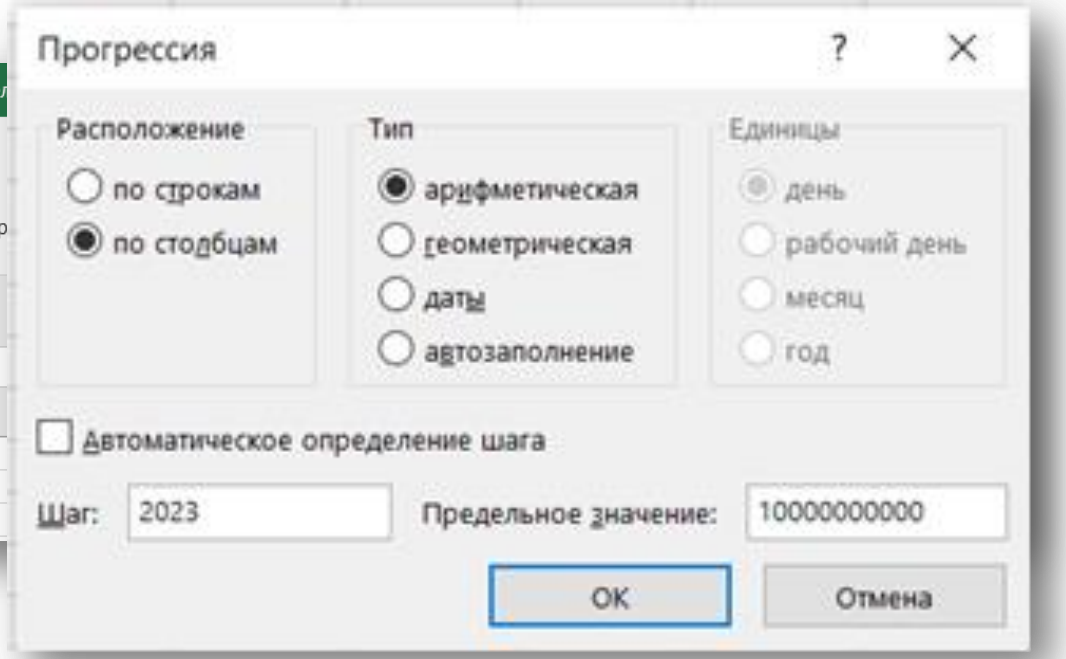


Решение (Таблицы, Excel, 3 способ)

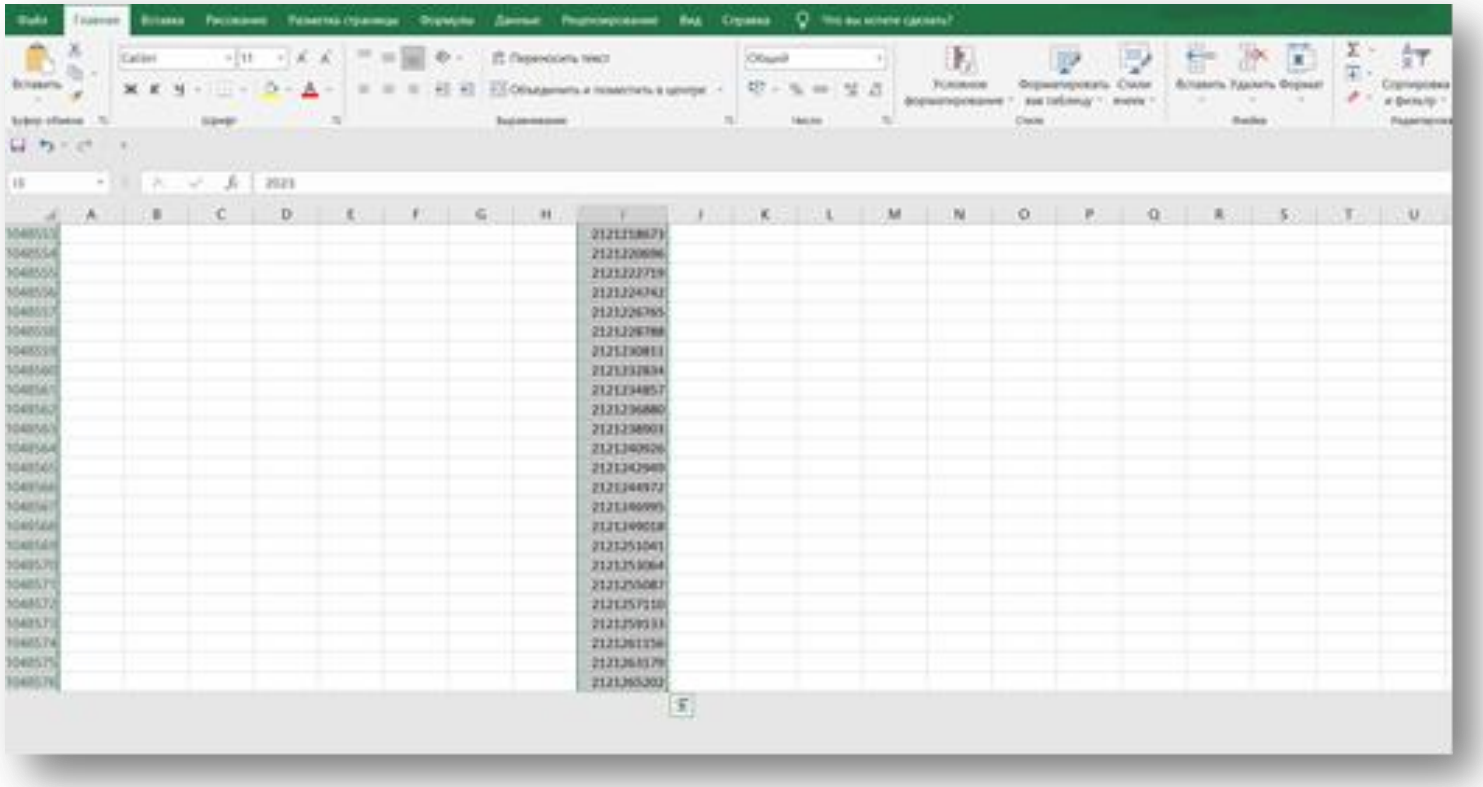
Сгенерируем последовательность из всех чисел, которые нам могут подойти (с помощью прогрессии). Так как числа должны делиться на 2023, то мы будем проверять не все 10^{10} чисел, а только кратные 2023: 2023, 4046, 6069 …

Выбираем расположение – «По столбцам», тип – арифметическая, шаг - 2023 и предельное значение – 1010.





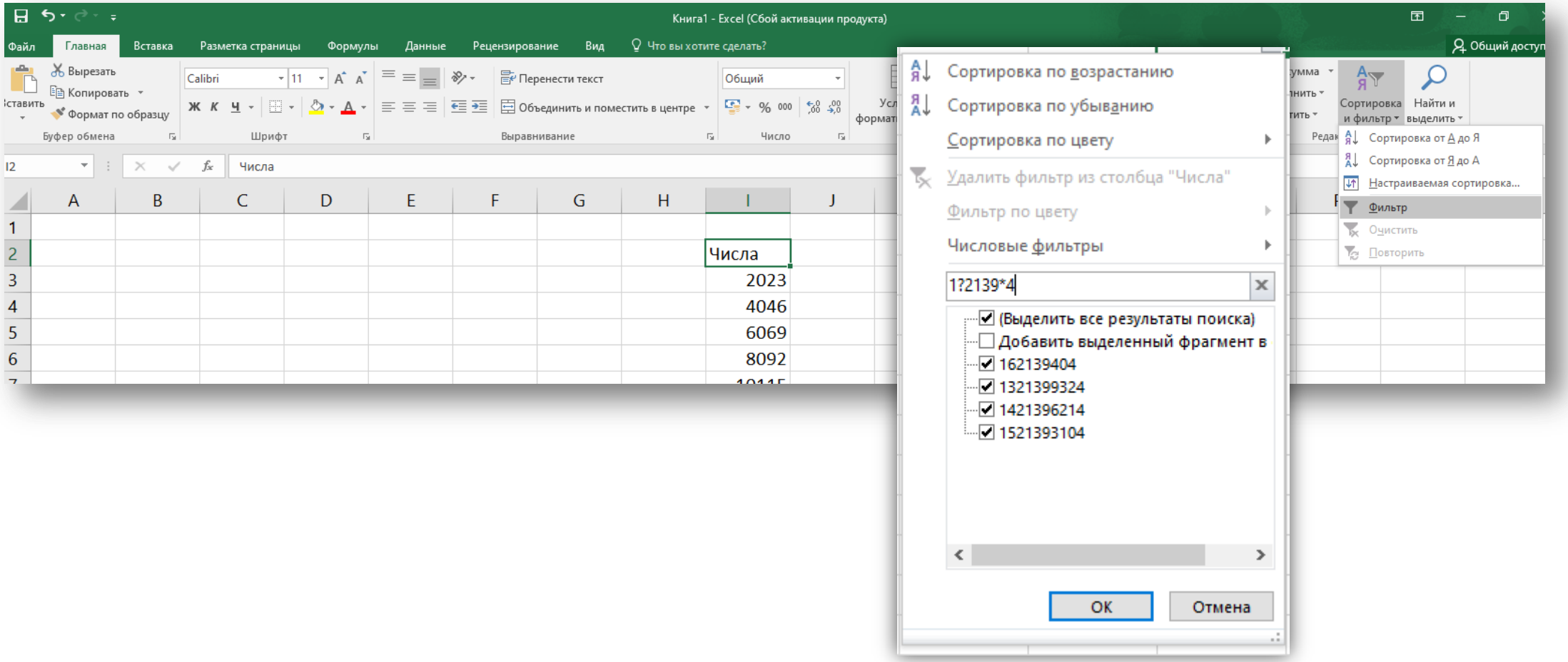
Решение (Таблицы, Excel, 3 способ)



Последовательность сгенерировалась. Нажимаем ***ctrl + shift + стрелка вниз***.
Все числа не поместились. Но учитывая маску **1?2139*4** и предельное значение 10^{10}, все числа, среди которых будем осуществлять поиск, сгенерировались (нужные нам числа начинаются с единицы и имеют не более десяти знаков)

Решение (Таблицы, Excel, 3 способ)

Применяем фильтр. В поле поиска вставляем нужную маску



Спецификация.

Задание 25 (высокий уровень)

- ✓ Обработка целых чисел.
- ✓ **Проверка делимости**

Необходимые навыки

- ✓ Проверка числа на простоту
- ✓ Поиск всех делителей
- ✓ Основная теорема арифметики

Поиск всех делителей

```
def allDivs( x ):
  divs = [1, x]
  d = 2
  while d*d <= x:
    if x % d == 0:
      divs.append( d )
      if x // d > d:
        divs.append( x//d )
    d += 1
  return sorted(divs)
```

Делители в парах: $x = d \cdot g \; (d \le g) \Rightarrow \; d \le \sqrt{x}$

Проблема: вещественное $\sqrt{x}$!

$$g = \frac{x}{d}$$

$$d \le \sqrt{x} \;\; \Rightarrow \;\; d^2 \le x$$

Проблема: полные квадраты!

$d = g$ $\Rightarrow$ один делитель

Проверка числа на простоту

```
def isPrime( x ):
  if x <= 1: return False
  d = 2
  while d*d <= x:
    if x % d == 0:
      return False
    d += 1
  return True
```

Пример

(Статград) Найдите все натуральные числа, принадлежащие отрезку [289123456; 389123456] и имеющие ровно три нетривиальных делителя. Для каждого найденного числа запишите в ответе его наибольший нетривиальный делитель.

Решение

```
def allDivs( x ):
  divs = []
  d = 2
  while d*d <= x:
    if x % d == 0:
      divs.append( d )
      if x // d > d:
        divs.append( x//d )
    d += 1
  return sorted(divs)

start = 289123456
end = 389123456
for x in range(start, end+1):
    if len(allDivs(x)) == 3:
        print(x)
```

долго считает…

? Как ускорить?

Основная теорема арифметики

Любое число единственным способом представляется в виде произведения простых чисел:

$$n = p_1^{k_1} p_2^{k_2} \dots p_m^{k_m}$$

Число нетривиальных делителей:

$$\delta = (k_1 + 1)(k_2 + 1)\dots(k_m + 1) - 2$$

Если $\delta = 3$:

$$(k_1 + 1)(k_2 + 1)\dots(k_m + 1) = 5$$

$$k_1 = 4,\ k_2 = k_3 = \dots k_m = 0$$

Решение

```
def isPrime(x):
    if x <= 1: return False
    d = 2
    while d * d <= x:
        if x % d == 0:
            return False
        d += 1
    return True


start = 289123456
end = 389123456
for x in range(int(start ** 0.25), int(end ** 0.25) + 1):
    if isPrime(x):
        print(x**4)
```