МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

ФГБОУ ВО «Пермский государственный

национальный исследовательский университет»

Физический факультет

Кафедра радиоэлектроники

и защиты информации

ОТЧЁТ

По лабораторной работе «Алгоритм квадратичного решета»

по дисциплине «Криптографические методы защиты информации»

Работу выполнил студент ФЗ-13

Заболотько Семён Ильич

**Постановка задачи**

1. Выбрать факторизуемое значение N
2. Через Алгоритм квадратичного решета вычислить p и q

**Краткая теория**

Метод квадратичного решета (Quadratic sieve algorithm, сокр. QS) — метод [факторизации](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) больших чисел, разработанный Померанцем в 1981 году. Долгое время превосходил другие методы факторизации целых чисел общего вида, не имеющих простых делителей, порядок которых значительно меньше sqrt(n) (для чисел n, имеющих простые делители, много меньшие sqrt(n) более быстрым является [метод факторизации на эллиптических кривых](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D0%9B%D0%B5%D0%BD%D1%81%D1%82%D1%80%D1%8B_%D1%81_%D0%BF%D0%BE%D0%BC%D0%BE%D1%89%D1%8C%D1%8E_%D1%8D%D0%BB%D0%BB%D0%B8%D0%BF%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D1%85_%D0%BA%D1%80%D0%B8%D0%B2%D1%8B%D1%85)). Метод квадратичного решета представляет собой разновидность [метода факторных баз](https://ru.wikipedia.org/w/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%BD%D1%8B%D1%85_%D0%B1%D0%B0%D0%B7&action=edit&redlink=1) (обобщение [метода факторизации Ферма](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8_%D0%A4%D0%B5%D1%80%D0%BC%D0%B0)). Этот метод считается вторым по быстроте (после [общего метода решета числового поля](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%89%D0%B8%D0%B9_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4_%D1%80%D0%B5%D1%88%D0%B5%D1%82%D0%B0_%D1%87%D0%B8%D1%81%D0%BB%D0%BE%D0%B2%D0%BE%D0%B3%D0%BE_%D0%BF%D0%BE%D0%BB%D1%8F)). И до сих пор является самым быстрым для целых чисел до 100 десятичных цифр и устроен значительно проще чем [общий метод решета числового поля](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%89%D0%B8%D0%B9_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4_%D1%80%D0%B5%D1%88%D0%B5%D1%82%D0%B0_%D1%87%D0%B8%D1%81%D0%BB%D0%BE%D0%B2%D0%BE%D0%B3%D0%BE_%D0%BF%D0%BE%D0%BB%D1%8F). Это универсальный алгоритм факторизации, так как время его выполнения исключительно зависит от размера факторизуемого числа, а не от его особой структуры и свойств.

**Идея Алгоритма:**

Алгоритм пытается найти такие квадраты чисел, которые [равны по модулю](https://en.wikipedia.org/wiki/congruence_of_squares) n(факторизуемое число), что часто приводит к факторизации n. Алгоритм работает в два этапа: этап сбора данных, где он собирает информацию, которая может привести к равенству квадратов; и этап обработки данных, где он помещает всю собранную информацию в [матрицу](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) и обрабатывает её для получения равенства квадратов. Первый этап может быть легко [распараллелен](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B) на много процессов, но второй этап требует большие объемы памяти и его трудно распараллелить.

Метод квадратичного решета является модификацией [метода разложения на множители Диксона](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%94%D0%B8%D0%BA%D1%81%D0%BE%D0%BD%D0%B0).

Продолжительность расчёта для квадратичного решета (n - факторизуемое число):



**Подход:**

Пусть x mod y обозначает остаток от деления x на y. В [методе факторизации Ферма](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8_%D0%A4%D0%B5%D1%80%D0%BC%D0%B0) в отдельности подбираем число a, чтобы a2 mod n являлось квадратом. Но такое число подобрать тяжело. В квадратичном решете мы вычисляем a2 mod n для некоторых a, и затем находим такое подмножество, произведение элементов которого является квадратом. Это приведёт к сравнению квадратов.

Например, 412 mod 1649 = 32, 422 mod 1649 = 115, и 432 mod 1649 = 200. Ни один из полученных результатов не является квадратом, но результат произведения (32)(200) = 6400 = 802. С другой стороны, рассмотрев предыдущее произведение по mod 1649, мы получим, что (32)(200) = (412)(432) = ((41)(43))2. Так как (41)∙(43) mod 1649 = 114, мы имеем сравнение квадратов: 1142 ≡ 802 (mod 1649).

Но как мы решаем проблему, фиксируя множество чисел и, находя подмножество, произведение элементов, которого является квадратом? Начнём с понятия [вектор](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D1%81%D1%82%D1%80%D0%B0%D0%BD%D1%81%D1%82%D0%B2%D0%BE) показателей степеней. Например, у нас есть число 504. Его разложение на простые множители имеет следующий вид 504 = 23325071. Мы могли бы представить это разложение в виде вектора показателей степеней (3,2,0,1), который фиксирует степени простых чисел 2, 3, 5 и 7, участвующих в разложении. Число 490 по аналогии имело бы вектор (1,0,1,2). Умножение чисел - это то же самое, что и поэлементное сложение их векторов показателей степеней, то есть произведение 504 ∙ 490 имеет вектор (4,2,1,3).

Теперь, обратите внимание, что число является квадратом, если каждый элемент в его векторе показателей степеней [чётный](https://ru.wikipedia.org/wiki/%D0%A7%D0%B5%D1%82%D0%BD%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE). К примеру, при сложении векторов (3,0,0,1) и (1,2,0,1) получаем (4,2,0,2). Это говорит нам о том, что произведение чисел 56 ∙ 126 является квадратом. На самом деле, мы не заботимся о точных значениях, получаемых в векторе, до тех пор, пока каждый элемент в результирующем векторе чётный. По этой причине мы берём каждый элемент по mod 2 и выполняем сложение элементов по mod 2: (1,0,0,1) + (1,0,0,1) = (0,0,0,0).

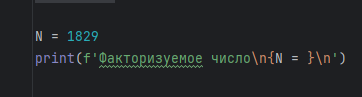
Таким образом, наша задача приняла следующий вид: задано множество векторов (0,1), найти такое подмножество, которое дополняется до нулевого вектора, при использовании сложения по mod 2. Это задача линейной алгебры, то есть необходимо найти линейно зависимые вектора. Из теоремы линейной алгебры следует, что, до тех пор, пока количество векторов больше, чем количество элементов в каждом векторе, такая зависимость должна существовать. Мы можем эффективно находить линейно зависимые векторы, например, поместив исходные векторы, в качестве столбцов матрицы и затем, использовать [метод Гаусса](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%93%D0%B0%D1%83%D1%81%D1%81%D0%B0), который легко приспособить для работы с целыми числами по mod 2. Как только мы найдём линейно зависимые векторы, мы просто перемножаем числа, соответствующие этим векторам и получаем квадрат, который ищем.

Однако, возведение в квадрат множества случайных чисел по mod n приводит к большому числу различных простых множителей,длинным векторам и большой матрице. Чтобы избавиться от этой проблемы, мы специально ищем числа, такие, что a2 mod n имеет только небольшие простые множители (такие числа называются [гладкими числами](https://ru.wikipedia.org/wiki/%D0%93%D0%BB%D0%B0%D0%B4%D0%BA%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE)). Их сложнее найти, но использование таких чисел позволяет избежать больших векторов и матриц.

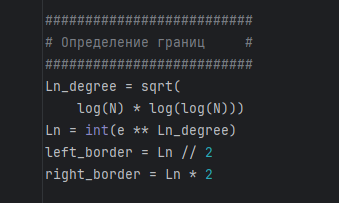
**Выполнение**

Наша задача – получить значения p q при известном n

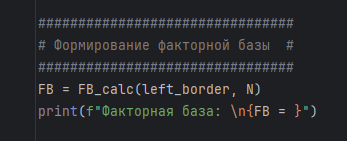
Возьмем n из примера с лекции

****

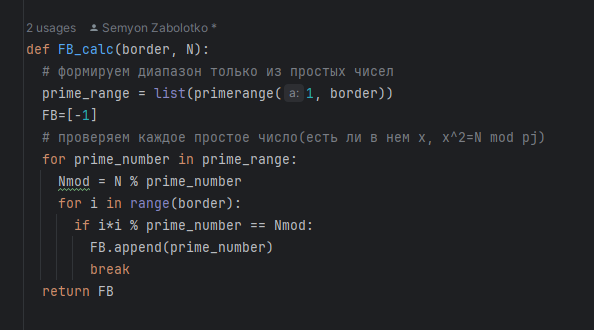
Определим границы с помощью значения L(n)



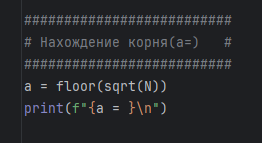
Теперь сформируем факторную базу



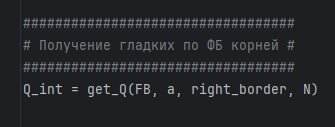
Она будет состоять из простых чисел, которые прошли отбор с нашим N



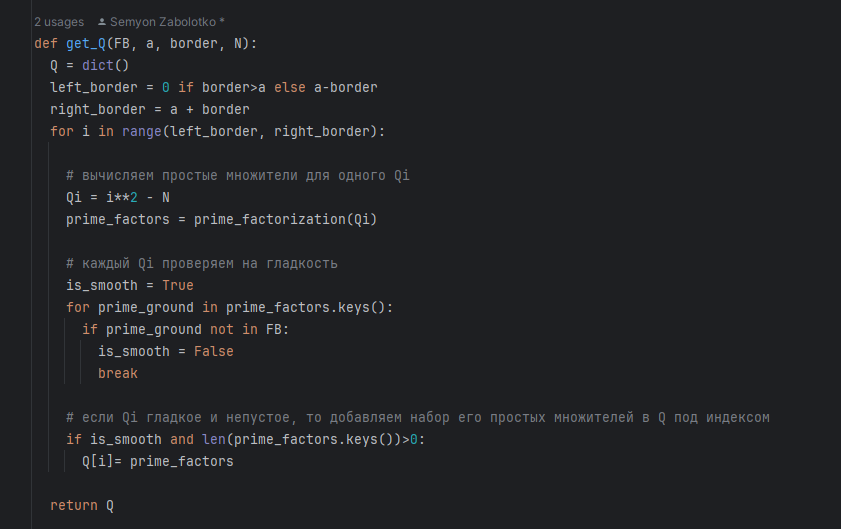
Далее найдём значение a



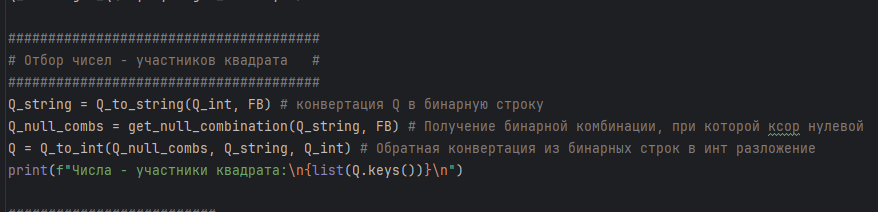
Теперь нужно получить гладкие по ФБ корни в виде словаря с индексами Q



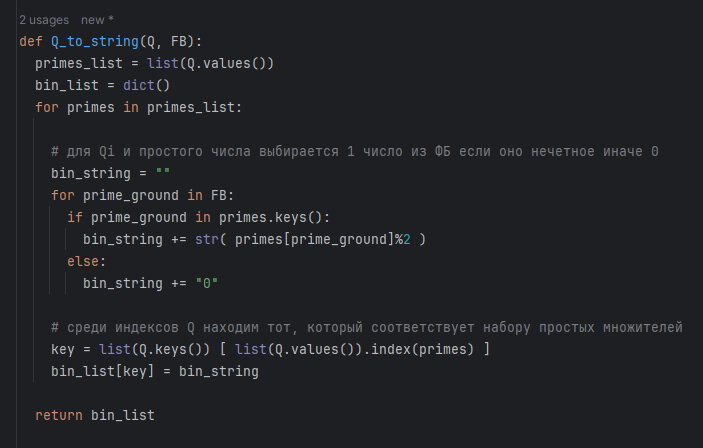
Все Q[i] находятся через ФБ, значение а, границу и число N



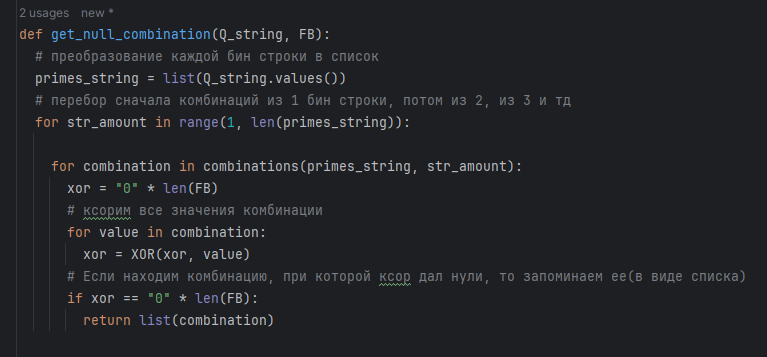
Далее идет отбор чисел – участников квадрата в несколько функций



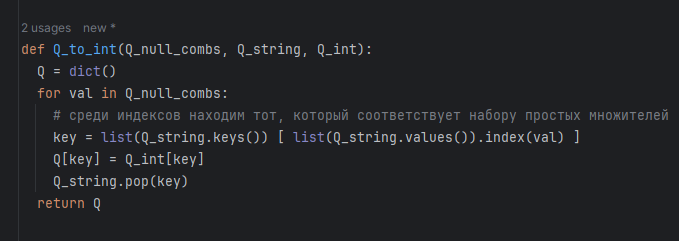
Сначала Q переводится в бинарный вид



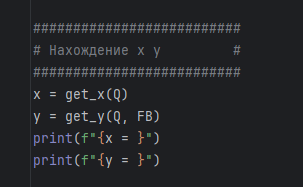
Далее вычисляются бинарные комбинации при ксоре которых выходит нулевое бинарное значение



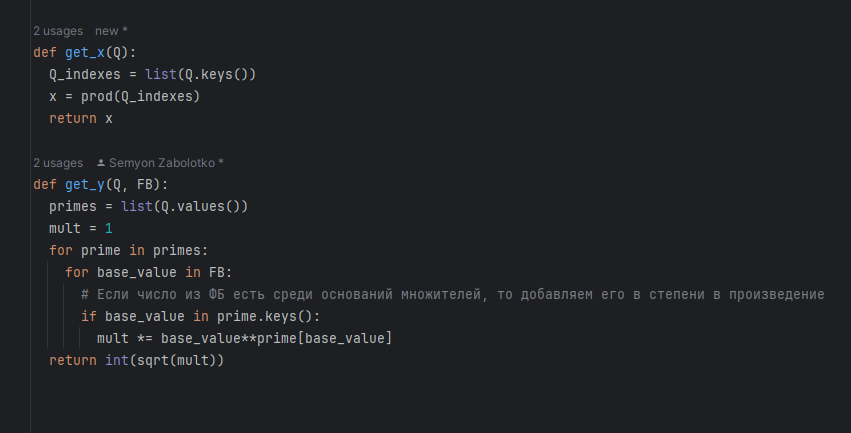
И в конце Q обратно переводится в инт



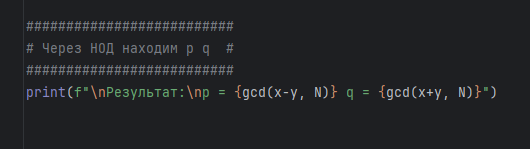
Теперь мы можем с помощью вычисленного Q найти x y

****

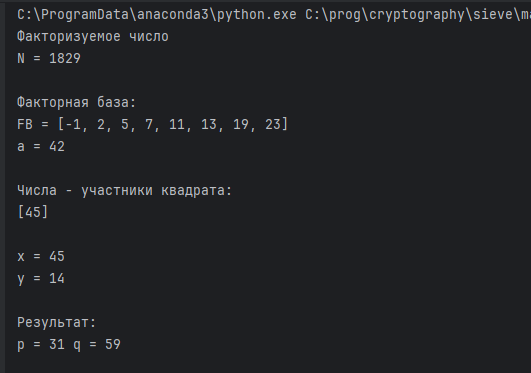
Они находятся через перемножение оснований и множителей соответственно

****

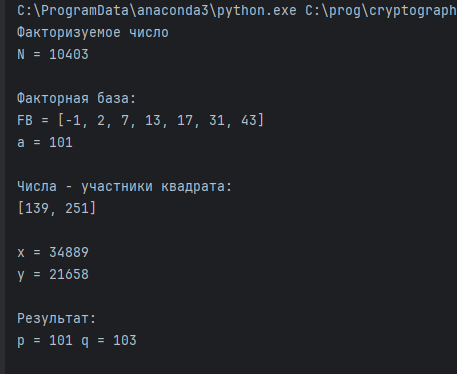
И теперь можно спокойно вычислить через НОД наши целевые p q



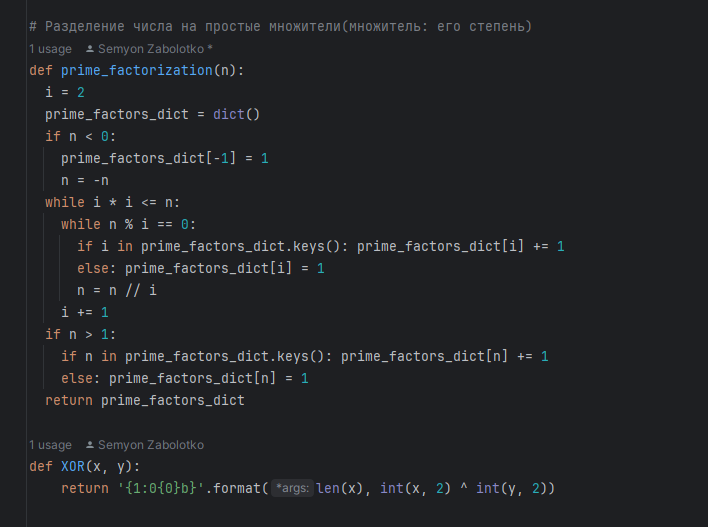
Полный консольный вывод:



Так же с N=10403



Доп функции(XOR и разложение на простые множители)

****

**Выводы**

Мы научились применять алгоритм квадратичного решета для вычисления степени p q при известном n. Решили задачу факторизации