Отчет лабораторной работы №1
По дисциплине “Криптография”
Выполнил Кондратьев Егор М8О-306Б-19

Вариант 14
(n1 – 77 чисел, n2 – 463 чисел)
n1=611219701749111463195451937544251195208759452152827846401772765239293765019 13
n2=623859693199013147827532715234380179966825776270558257642799781597622030920769121143520500490376729017323087214130721302939229640132476986634867943474326596357137590234739434371411059004396261781732670972951818034845228440270705576583264892500062621359605386517311623860359205198607332952890850212529591183712450115109734560508215282728727996124015076658960966561467552701222936398203576600544336692557486353756923598897702847555993246278174277173208452762972207 1

В олимппроге часто встречаются задачи по теории чисел. И кстати задача разложения числа на простые множители одна из самых частых. Поэтому я, конечно, знаю асимптотику алгоритмов для решения таких задач и это в основном: O(n * log n) и O(sqrt n). Их, конечно, достаточно для решения контестов, но для выполнения данной лабки они будут так себе…. мягко сказано….хех.

Поэтому я начал искать эффективные методы решения и нашел - msieve(факторизация больших чисел с помощью квадратичного решета).
Мне не понравилось его устанавливать, так как это какие-то танцы с бубном получались, поэтому тупо нашел онлайн-инструмент:
https://www.cryptool.org/en/cto/msieve

В итоге за 2 минуты, ответ на первое число был найден:
p = 212453377902490714807152598462337631023
q = 28769592076320937531086846262550219743 1
что прошло мою нехитрую проверку

```
du@Du:~$ python3
Python 3.8.10 (default, Sep 28 2021, 16:10:42)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> p = 212453377902490714807152598462337631023
>>> q = 28769592076320937531086846262550219743 1
>>> p*q
611219701749111463195451937544251195208759452152827846401772765239293765019 13
>>>
```

Второе число сайт не смог считать из-за ограничения на длину. Даже если не учитывать ограничение на длину, сайт использовать всё равно нет смысла, так как алгоритм в среднем потратил бы более 20 часов на факторизацию второго числа…к чему я конечно не готов.

Здесь я понимаю, тут не обойдется без НОДов и чтобы не пытаться рандомить, можно воспользоваться числами n2 из других вариантов, а чтобы было больше шансов, можно проверить их все.
И, если найдется НОД, будем считать его первым нашим ответом q. А чтобы найти второе p, мы можем бесхитростно поделить наше n2 на НОД, и ТАДАМ. Но это пока теория, осталось реализовать. Что явно будет быстрым.

```
from math import gcd

num14 =
623859693199013147827532715234380179966825776270558257642799781597622030920769121143520500490376729017323087214130721302939229640132476986634867943474326596357137590234739434371411059004396261781732670972951818034845228440270705576583264892500062621359605386517311623860359205198607332952890850212529591183712450115109734560508215282728727996124015076658960966561467552701222936398203576600544336692557486353756923598897702847555993246278174277173208452762972207 1

with open('n.txt') as f:
    for line in f:
        if line[0] != 'n':
            var = line[0: -1]
        else:
            num = int(line[3: -2])
            gcd_n = gcd(num14, num)
            if num == num14:
                print("Одно и тоже")
            elif gcd_n != 1:
                tmp = num14 // gcd_n
                print("Вариант {}, {} -- Сопряженное число, подошло".format(var, line[0: 2]))
                print("n{} = {}\n".format(line[1], num))
                print("q = {} (digits): {}\n".format(len(str(gcd_n)), gcd_n))
                print("p = {} (digits): {}\n".format(len(str(tmp)), tmp))
                check = tmp * gcd_n
                print("Нехитрая проверка\nq * p = {}".format(check))
                exit()
            else:
                print("Вариант {}, {} : q = {} - печально,это нам не подходит(".format(var, line[0: 2], gcd_n))
```

C:/Users/egork/Desktop/space/crypto/lab1/main.py
Вариант 0, n1 : q = 1 - печально, это нам не подходит(
Вариант 0, n2 : q = 1 - печально, это нам не подходит(
Вариант 1, n1 : q = 1 - печально, это нам не подходит(
Вариант 1, n2 : q = 1 - печально, это нам не подходит(
Вариант 2, n1 : q = 1 - печально, это нам не подходит(
Вариант 2, n2 : q = 1 - печально, это нам не подходит(
Вариант 3, n1 : q = 1 - печально, это нам не подходит(
Вариант 3, n2 : q = 1 - печально, это нам не подходит(
Вариант 4, n1 : q = 1 - печально, это нам не подходит(
Вариант 4, n2 : q = 1 - печально, это нам не подходит(
Вариант 5, n1 : q = 1 - печально, это нам не подходит(
Вариант 5, n2 : q = 1 - печально, это нам не подходит(
Вариант 6, n1 : q = 1 - печально, это нам не подходит(
Вариант 6, n2 : q = 1 - печально, это нам не подходит(
Вариант 7, n1 : q = 1 - печально, это нам не подходит(
Вариант 7, n2 : q = 1 - печально, это нам не подходит(
Вариант 8, n1 : q = 1 - печально, это нам не подходит(
Вариант 8, n2 : q = 1 - печально, это нам не подходит(
Вариант 9, n1 : q = 1 - печально, это нам не подходит(
Вариант 9, n2 : q = 1 - печально, это нам не подходит(
Вариант 10, n1 : q = 1 - печально, это нам не подходит(
Вариант 10, n2 : q = 1 - печально, это нам не подходит(
Вариант 11, n1 : q = 1 - печально, это нам не подходит(
Вариант 11, n2 : q = 1 - печально, это нам не подходит(
Вариант 12, n1 : q = 1 - печально, это нам не подходит(
Вариант 12, n2 : q = 1 - печально, это нам не подходит(
Вариант 13, n1 : q = 1 - печально, это нам не подходит(
Вариант 13, n2 : q = 1 - печально, это нам не подходит(
Вариант 14, n1 : q = 1 - печально, это нам не подходит(
Одно и тоже
Вариант 15, n1 : q = 1 - печально, это нам не подходит(
Вариант 15, n2 -- Сопряженное число, подошло
n2 =
91731081875352815171407621167003846123266245546191597561703271310757656635924281843424981081668449878754721923794254026172615377510174617891758112810662901121449954857719282776786450462685156058364136389154080972201881402751800893107343052551388864437499966122341170119119045726872737908981849869847860123093368219862117195086830699797355493220157070350163979612771893561720282005021433241544281839262135063374958410354786680654395428194800001049496666864308342553

q = 155 (digits):
260284607242970468044510054235008830881986286517101086505068966897557925709195767286319862958048786343846817976462962978186096019263768018363563340185306 19

p = 309 (digits):
239683667738619909435043761168696430649299798361354713695469181019038516148277464441152469153963503274525422864716814088783604510284591969867575321185053713679626766373510036682383305698495477101578286136263535103041609053068820761949764530011524783621969538707570492517608979687945702684287013188139386235 09

Нехитрая проверка
q * p =
623859693199013147827532715234380179966825776270558257642799781597622030920769121143520500490376729017323087214130721302939229640132476986634867943474326596357137590234739434371411059004396261781732670972951818034845228440270705576583264892500062621359605386517311623860359205198607332952890850212529591183712450115109734560508215282728727996124015076658960966561467552701222936398203576600544336692557486353756923598897702847555993246278174277173208452762972207 1

Process finished with exit code 0

И да, получилось, странный способ, конечно, но работает, а значит трогать не надо.

в итоге получились такие ответы:

N1:

1. p = 21245337790249071480715259846233763102 3
2. q = 28769592076320937531086846262550219743 1

N2:

1. q = 155 (digits):
   260284607242970468044510054235008830881986286517101086505068966897557925709195767286319862958048786343846817976462962978186096019263768018363563340185306 19
2. p = 309 (digits):
   239683667738619909435043761168696430649299798361354713695469181019038516148277464441152469153963503274525422864716814088783604510284591969867575321185053713679626766373510036682383305698495477101578286136263535103041609053068820761949764530011524783621969538707570492517608979687945702684287013188139386235 09