

# Отчёт по лабораторной работе №2 по курсу «Криптография»

Выполнил Попов Николай, группа М8О-308Б-21

## Задание

Разложить каждое из чисел  $n_1$  и  $n_2$  на нетривиальные сомножители. Ниже представлены 40 вариантов.

Вариант 16:

$n_1 =$

3834566148849024667262527312945442346580153906193728358  
26246625499154384118189

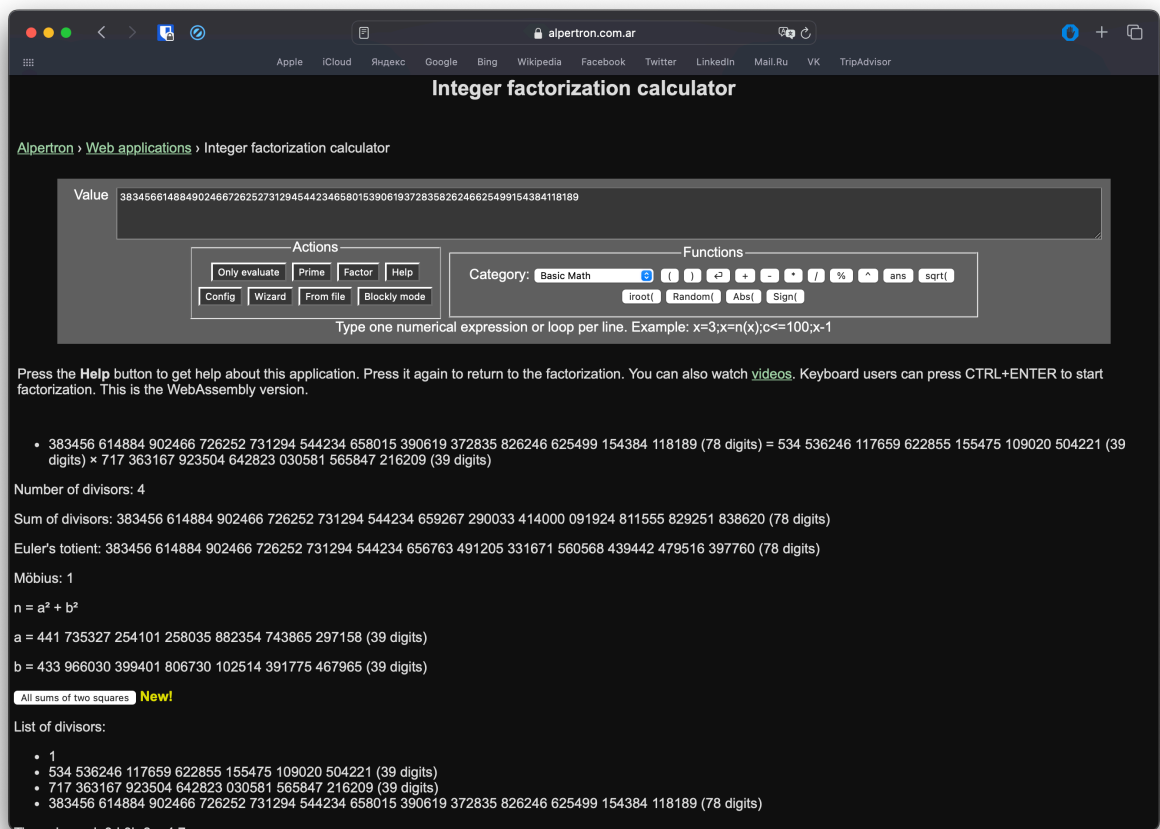
$n_2 =$

1416908444771934114327236064335695175033855568724514723  
2760909092389022494506116311617929837009763773660959874  
6978539681190806175023739437824949790203114195447287621  
1921620528639113700302812533115824770238590279848186791  
0823926760076341189111357818193897834136876367785553468  
5413427437290239276573078365437316891195505584463642669  
7161129367283730885533859028643592189337506274405214704  
767741787934130977543281068768100090

## Ход работы

Для разложения первого числа я попытался сгенерировать достаточно много простых чисел в надежде что среди них найдется хотя бы один делитель моего числа. Проверив первый миллион простых чисел я обнаружил, что среди них нет делителей моего числа из условия. Генерация более миллиона простых чисел требовала значительно большего времени, поэтому для факторизации первого числа я прибегнул к онлайн-инструментам, использующим продвинутые алгоритмы разложения чисел на простые сомножители.

При помощи алгоритма ECM на сайте [www.alpertron.com.ar/ECM.HTM](http://www.alpertron.com.ar/ECM.HTM) мне удалось примерно за 7 минут разложить первое число на простые сомножители



Разложение первого числа:

$$n1 = p * q$$

$$p = 534536246117659622855155475109020504221$$

$$q = 717363167923504642823030581565847216209$$

По длине этих чисел можно заметить, что простой перебор делителей или генерация простых чисел при помощи стандартных методов, таких как решето Эратосфена не помогут нам решить поставленную задачу в разумное время.

Я попробовал поискать НОД своего числа и чисел из остальных вариантов и мне удалось найти число, НОД с которым был отличен от единицы. Полученный НОД - первый множитель разложения. Второй множитель - мое число, деленное на этот НОД

```
import math

# числа n1 из вариантов
small_numbers = []

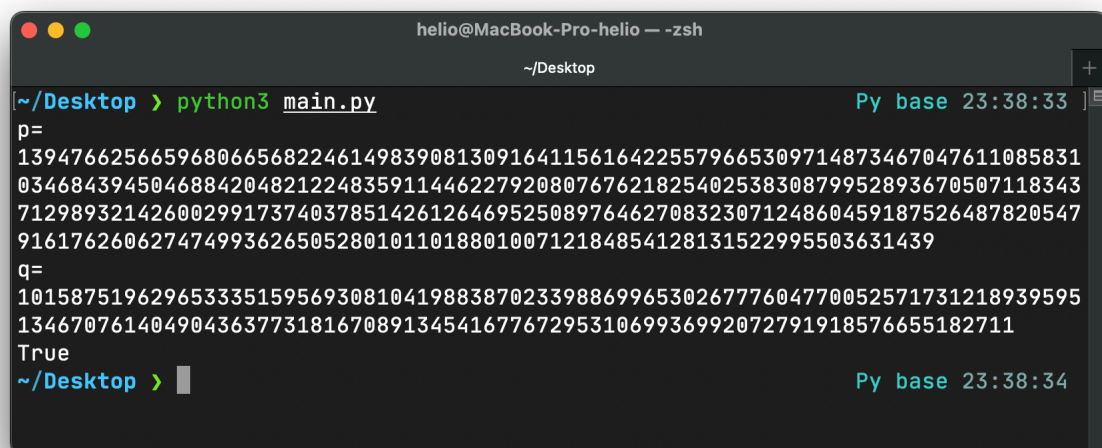
# числа n2 из вариантов
big_numbers = []

number_to_factor =
14169084447719341143272360643356951750338555687245147232760909092389022494507611631161
79298370097637736609598746978539681190806175023739437824949790203114195447287621192162
```

```
05286391137003028125331158247702385902798481867910823926760076341189111357818193897834
13687636778555346854134274372902392765730783654373168911955055844636426697161129367283
73088553385902864359218933750627440521470476774178793413097754328106876810009083432628
213288672194420754620920548851129
```

```
for num in small_numbers + big_numbers:
    gcd = math.gcd(num, number_to_factor)
    if gcd != 1 and gcd != number_to_factor:
        p, q = gcd, number_to_factor // gcd

print("p=")
print(p)
print("q=")
print(q)
print(p * q == number_to_factor)
```



```
helio@MacBook-Pro-helio -- zsh
~/Desktop
[~/Desktop > python3 main.py Py base 23:38:33]
p=
13947662566596806656822461498390813091641156164225579665309714873467047611085831
03468439450468842048212248359114462279208076762182540253830879952893670507118343
71298932142600299173740378514261264695250897646270832307124860459187526487820547
916176260627474993626505280101101880100712184854128131522995503631439
q=
10158751962965333515956930810419883870233988699653026777604770052571731218939595
134670761404904363773181670891345416776729531069936992072791918576655182711
True
~/Desktop > Py base 23:38:34
```

Таким образом разложение  $n_2$ :

$$n_1 = p * q$$

```
p =
1394766256659680665682246149839081309164115616422557966530971487346704
7611085831034684394504688420482122483591144622792080767621825402538308
7995289367050711834371298932142600299173740378514261264695250897646270
8323071248604591875264878205479161762606274749936265052801011018801007
12184854128131522995503631439

q =
1015875196296533351595693081041988387023398869965302677760477005257173
1218939595134670761404904363773181670891345416776729531069936992072791
918576655182711
```

## Вывод

Проделав лабораторную работу, я на собственном опыте убедился, что крайне сложно найти нетривиальные сомножители большого числа, а значит такие разложения могут успешно использоваться в операциях шифрования и дешифрования в системах с открытым ключом.