**Отчет по лабораторной работе №1 по курсу «Криптография»**

Выполнил Моисеенков Илья Павлович, М8О-308Б-19.

***Задание***

Разложить каждое из чисел n1 и n2 на нетривиальные сомножители.

n1 = 374456902508739435218273258671224457341348406488533188195528827819627513233269

n2 = 6388532302085669228615771388983452948007941743568163970560946831157382344058239126229806052859737888487783937846720963023038153065348279670854149017787470574812006836885115106891910821052589226052685423814611520013606076441149759993153875258431039836687849147729558354065212066523592281254609943962306220641867429175693807391177713424759495700540370707278314153079064977575088510104373434672731111688909374407992653018176506344174461412097021874839013850305001081

***Ход работы***

Для разложения первого числа я воспользовался готовой онлайн реализацией метода квадратичного решета. Этот метод считается вторым по быстроте (после общего метода решета числового поля). И до сих пор является самым быстрым для целых чисел до 100 десятичных цифр и устроен значительно проще чем общий метод решета числового поля. Разложение первого числа (78 разрядов) заняло примерно 10 минут.

Результат:

n1 = p1 \* q1

p1 = 573733447260765268493955136095177485171

q1 = 652667025596202591708982935779112779639

Со вторым числом все оказалось гораздо интереснее. Оно состоит из 463 разрядов. Ради интереса я попробовал факторизовать это число тем же методом, но спустя сутки я не получил никакого результата. Я посмотрел на много реализаций различных алгоритмов, но ни одна из них не смогла разложить мое число даже спустя сутки.

Воспользовавшись подсказкой, что числа из вариантов строятся по похожей схеме, я решил прибегнуть к хитрости и поискать НОД своего числа и чисел из остальных вариантов. К счастью, мне удалось найти число, НОД с которым был отличен от единицы. Полученный НОД - первый множитель разложения. Второй множитель - мое число, деленное на этот НОД.

Результат:

n2 = p2 \* q2

p2 = 20591237055704179681625834644880484498444984995772117549539485534190132602467078164211345327341051193704702361911450307821215708165717411436692397336069833

q2 = 310254905268837138750560033553835447082078617634065472300263027998037659479137610182632126068189456517307258215912470072567391420497863505202627004364772496597552620989779445924457883982777253633502440214082777799289608925711332037147354274920805306328865963901630631044266853074063760442778730864597255838257

***Код***

Приведу код, который помог мне в нахождении НОД.

*NUMS = [] # все числа из других вариантов (в отчет не вставлял)*

*MY\_NUM = n2 # число n2 моего варианта*

*import* math  
  
*for* num *in* NUMS:  
 gcd = math.gcd(num, MY\_NUM)  
 *if* gcd != 1 *and* gcd != MY\_NUM:  
 *print*(MY\_NUM)  
 *print*('=')  
 *print*(gcd)  
 *print*('\*')  
 *print*(MY\_NUM // gcd)

***Выводы***

Изначально работа показалась мне максимально простой. Я рассчитывал на то, что мне удастся найти какой-нибудь маленький делитель и разложить число на два множителя. Но проблема оказалась в том, что эти числа довольно огромные. И получены они перемножением двух больших простых чисел. Поэтому тривиальные алгоритмы не сработали.

Мне пришлось почитать про другие алгоритмы факторизации, которые работают для больших чисел. С помощью готовой реализации одного из таких алгоритмов - метода квадратичного решета - я смог получить разложение первого числа за 10 минут.

Второе число, как мне показалось, вообще почти нереально разложить на множители за приемлемое время. Задание мне удалось выполнить только прибегнув к хитрости.

Но благодаря этому заданию я понял, почему в криптографии так часто фигурируют простые числа - перемножив всего лишь два больших простых числа, мы получаем довольно большое и почти не факторизуемое число, которое можно использовать в различных алгоритмах шифрования.