Лабораторная работа №3

Тема: Криптосистема RSA.

Введение. Формально, как и для любой ассиметричной криптосистемы, для работы RSA необходимо реализовать 3 основные процедуры: генерацию ключей, функцию зашифрования и функцию расшифрования. Подробности можно посмотреть в лекции.

Генерация ключей (Gen).

1. Выбираются два больших простых числа p и q.
2. Вычисляется число $n = pq$; n –модуль.
3. Открытый ключ e выбирается из $Z_{\phi(n)} = \{0, \ldots, \phi(n)-1\}$ таким образом, чтобы $НОД\ (e, \phi(n)) = 1$ (очевидно, что зная p и q, можно легко вычислить $\phi(n) = (p-1)(q-1)$).
4. Личный ключ d из $Z_{\phi(n)}$ находится по формуле $d = e^{-1} \pmod{\phi(n)}$.

Функция зашифрования (Encr).

$E_e(X) = X^e \pmod n$, где n –модуль, e – открытый ключ.

Функция расшифрования (Decr).

$D_d(Y) = Y^d \pmod n$, где n –модуль, d –личный ключ.

Замечания:

1) Для возведения в степень рекомендуется использовать алгоритм быстрого возведения в степень. Нельзя использовать готовую реализацию этого алгоритма, если она есть в языке программирования, на котором вы выполняете лабораторную работу.
2) Проверить, что случайно выбранное число e удовлетворяет требуемому условию $НОД(e, \phi(n)) = 1$, и, если условие выполнено, найти личный ключ $d = e^{-1} \pmod{\phi(n)}$ можно с помощью расширенного алгоритма Евклида. Нельзя использовать готовую реализацию этого алгоритма, если она есть в языке программирования, на котором вы выполняете лабораторную работу.
3) Сообщение X, которое необходимо зашифровать, и сообщение Y, которое необходимо расшифровать, представляются в виде некоторого числа из Z_n.

Условие лабораторной работы:

1) Для выполнения всех заданий необходимо использовать "длинную" арифметику. Разрешается использовать любую готовую библиотеку или написать свою.
2) Ваша программа должна принимать на вход указание, какую операцию необходимо выполнить: *Gen*, *Encr*, *Decr или Exit*. В зависимости от операции программа должна считывать (из консоли, файла и т.п.) необходимые данные:
    a. *Gen*: p, q и e;

b. *Encr*: *X*, *e* и *n*;
c. *Decr: Y*, *d* и *n:*
d. *Exit*: нет параметров.

3) На выход (в консоль, файл и т.п.) программа возвращать результат выполненной операции: *d* для *Gen*, *Y* для *Encr*, *X* для *Decr.* При вызове операции *Exit* программа должна завершать свою работу.

4) Программа должна иметь проверку корректности входных данных (для *Gen,* что НОД(e, $\phi(n)$) = 1; для *Encr* и *Decr,* что *X* и *Y* соответственно лежат в Z_n).

5) Программа должна предлагать пользователю выполнить очередную операцию до тех пор, пока пользователь не выберет вариант выхода из программы.

*Отчет (*условия по вариантам смотрите на последней странице)*:*

1) В условиях своего варианта для заданных чисел *p*, *q* и *e*, необходимо вычислить личный ключ *d*.
2) Для заданного сообщения *X*1, вычислить зашифрованное сообщение *Y*1, используя открытый ключ *e*.
3) Расшифровать сообщение *Y*1, используя личный ключ *d*, сравнить результат с исходным сообщением *X*1.
4) Для заданного шифртекста *Y*2, вычислить исходный открытый текст *X*2, используя личный ключ *d*.
5) Результаты шагов 1 – 4 необходимо записать в текстовый файл, назвав его «Report.txt».

Бонусные задания.

1) Реализовать один из вероятностных методов проверки числа на простоту:
a) Тест Соловея-Штрассена;
b) Тест Миллера- Рабина.

Данные тесты, проверяют, является ли число составным. Если хотя бы на одной итерации тест скажет, что число составное, то оно действительно составное, если же все итерации пройдут, то с вероятностью близкой к 1 (зависит от теста и числа итераций) можно утверждать, что число простое. Умея проверять число на простоту легко генерировать относительно большие простые числа: выбираем случайное нечетное число *n* требуемого нам размера (в битах), проверяем его на простоту, если оно простое, то все хорошо, если же нет, то берем число *n*+2 и т.д.

Реализация должна принимать на вход желаемый размер простого числа (в битах), а на выходе возвращать простое (с вероятностью не ниже 0.999999) число заданного размера. Алгоритм должен использовать "длинную" арифметику.

Замечание: Бонусные задания не являются обязательными. Они дают возможность получить дополнительные баллы.

Порядок сдачи лабораторной работы:

1) Вам необходимо создать архив формата «.zip», название которого должно иметь вид «4_5_Ivanov.zip», где «5» – № вашей группы, а «Ivanov» – ваша фамилия латинскими буквами. В архиве должен быть исходный код вашей реализации и отчет по лабораторной работе.
2) Дедлайн: 22.10.2024.
3) Присылать задание на почту lolita.harmatnaya@gmail.com с пометкой «Л.Р. № 4 Группа Фамилия»

Таблица №1: Тестовые варианты параметров алгоритма RSA

№ вар.	*p*	*q*	*e*	*X*1	*Y*2
1	563036103490583	1063300642915937	372585779765210097553647509959	399754188907643924420059310699	293314580135454643114146935352
2	684391453787369	938396705691661	245372344253915653531369256899	184712154522842417799563173273	447204864183801463638208868116
3	950226133300007	973747816218557	272205786540380931859823391349	487590396324873679144487947752	371209390170967767404608751313
4	882493304303057	565640080106113	435510454193522616856570224823	33938304564942541056706890572	167669363821217143128176537107
5	801410357975153	950867021741191	110066171603901969362593059313	651256495894733822754552878879	253970845268857814403399216528
6	877624729981871	977289103272413	393858705445953556259457155603	992136842761319041440635938 42	33236459429354479423716278 41
7	578569278720973	976534805568533	235108486320061234453015373083	402700874043636335474593885222	416593343738152120120255791792
8	1102914252601991	571301412050021	624840313709071966800768010501	267222621555915275276288463243	291064433434228628162063527294
9	1044779951622553	810317495045789	657311140049004998195550456065	543066057187844905915287549311	482438945755863228767362999535
10	749491517671339	1074991257344881	224872067131777233484376270407	469827559223869083025116205997	135039041969537192266152584876
11	1003636526287921	630114564355681	227175758417350578804323023499	943804553322056696203178005 75	493089842126272405947077785493
12	630771436373581	786871162464589	145903195239272744119599032753	153836541207562724815850112591	154774251062114172643840396063
13	1101233370547069	913071788602213	441294907009469893617176298995	832192044845038413443817859011	381868705201087633178499417569
14	804288300171659	819139104388459	587862009679843002844824189377	201229993267158788910642144722	474981332560572636448191786787
15	722227767914309	645306156219341	188343213856435087990117867713	340693830559923670446313718411	401641252598150824512742688568