Дисциплина «Алгоритмы решения прикладных задач»
Рабочая тетрадь 8.
Шифрование с открытым ключом. Расширенный алгоритм Евклида. Алгоритм шифрования RSA

Теоретический материал

1) Расширенный алгоритм Евклида

Расширенный алгоритм Евклида — это расширение алгоритма Евклида, которое вычисляет, кроме наибольшего общего делителя (НОД) целых чисел **a** и **b**, ещё и коэффициенты соотношения Безу, то есть целые **x** и **y**, такие что

$$ax + by = \text{НОД}(a, b).$$

Пусть **d** - наибольший общий делитель чисел **a** и **b**. Тогда выражение **ax+by** всегда кратно **d**. Оказывается, что можно подобрать такие числа **x** и **y**, что **ax+by=d**. Эту задачу решает расширенный алгоритм Евклида. Рассмотрим его рекурсивную реализацию. Пусть функция gcdex получает на вход числа **a** и **b** и возвращает кортеж из трех чисел **d**, **x**, **y**, где **d** - наибольший общий делитель **a** и **b**, а **x** и **y** - такие целые числа, что **ax+by=d**.
Условием окончания рекурсии является **b=0**. В этом случае **d=a, x=1, y=0**. Если же **b≠0**, то вызовем функцию рекурсивно для чисел **b** и **a%b** и получим ответ для исходных чисел.

2) Нахождение обратного элемента по модулю.

Задача вычисления обратного элемента: по данным числам x, n, найти такое число x^{-1}, что $x \cdot x^{-1} \equiv 1 \pmod{n}$. Сперва запускается расширенный алгоритм Евклида для (n, x). Он выдаёт тройку $(1, i, j)$, где $in + jx \equiv 1 \pmod{n}$. Отсюда следует $jx \equiv 1 \pmod{n}$, и потому $j = x^{-1}$.

Пример 2. *Пример: вычислить* 3^{-1} *по модулю* 35. *Запускается расширенный алгоритм Евклида для* $(35, 3)$, *он выдаёт* $(1, -1, 12)$. *Ответ:* 12; *и верно,* $3 \cdot 12 \equiv 1 \pmod{35}$.

3) Возведение в степень по модулю

Чтобы вычислить x^y % N, нужно перемножить те степени x, которые соответствуют ненулевым позициям в двоичной записи y. Например,

$25_{10} = 11001_2$

$$x^{25} = x^{2^4} \cdot x^{2^3} \cdot x^{2^0} = x^{16} \cdot x^8 \cdot x^1$$

```cpp
#include <iostream>
using namespace std;
int modexp(int x, int y, int N)
{
  if (y == 0) return 1;
  int z = modexp(x, y / 2, N);
  if (y % 2 == 0)
    return (z*z) % N;
  else
    return (x*z*z) % N;
}
int main()
{
  int x, y, N;
  cout << "x= "; cin >> x;
  cout << "y= "; cin >> y;
  cout << "N= "; cin >> N;
  cout << modexp(x, y, N);
  cin.get(); cin.get();
  return 0;
}
```

4 .Алгоритм RSA

4) Создание открытого и секретного ключа

RSA-ключи генерируются следующим образом:

1) выбираются два различных случайных простых числа p и q заданного размера (например, 1024 бита каждое);

2) вычисляется их произведение $n = p \cdot q$, которое называется модулем;

3) вычисляется значение функции Эйлера от числа n:
$\varphi(n) = (p-1)\cdot(q-1)$;

4) выбирается целое число
$1 < e < \varphi(n)$, взаимно простое со значением функции $\varphi(n)$;
число e называется открытой экспонентой (англ. public exponent);
обычно в качестве e берут простые числа, содержащие небольшое количество единичных бит в двоичной записи, например,
простые из чисел Ферма: 17, 257 или 65537, так как в этом случае время, необходимое для шифрования с использованием
быстрого возведения в степень, будет меньше;

слишком малые значения e, например 3, потенциально могут ослабить безопасность схемы RSA.

5) вычисляется число

d, мультипликативно обратное к числу e по модулю $\varphi(n)$, то есть число, удовлетворяющее сравнению:

$$d \cdot e \equiv 1 \pmod{\varphi(n)}$$

(d называется секретной экспонентой; обычно оно вычисляется при помощи расширенного алгоритма Евклида);

6) пара (e, n) публикуется в качестве открытого ключа RSA (англ. RSA public key);

7) пара (d, n) играет роль закрытого ключа RSA (англ. RSA private key) и держится в секрете.

5) Шифрование

- Взять *открытый ключ (e, n)*
- Взять *открытый текст* m
- Зашифровать сообщение с использованием открытого ключа
  - $c = E(m) = m^e \mod n$

6) Расшифрование

- Взять *шифрованный текст c*
- Взять *закрытый ключ (d, n)*
- Применить закрытый ключ для расшифрования сообщения^

  $m = D(c) = c^d \mod n$

Задание 1
Напишите программу для реализации расширенного алгоритма Евклида
Решение:
Ответ:
Задание 2
Напишите программу для шифрования сообщения алгоритмом RSA (с генерацией открытого и секретного ключей)
Решение:
Ответ: