БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра информатики

Факультет КСиС

Специальность ИиТП

Индивидуальная практическая работа № 2

по дисциплине «Методы защиты информации»

Выполнил: Реут М.В.

группа 893551

Минск 2022

# **1. Постановка задачи**

**Цель**: реализовать на языке программирования C++ алгоритм

асимметричного шифрования RSA.

**Результат**: Программа шифрования, осуществляющая криптографическое

преобразование введенного текста с помощью алгоритма симметричного

шифрования RSA.

**Общая постановка задачи**: Создать программу, читающую данные из

файла, шифрующие и дешифрующие их с помощью алгоритма RSA.

# **2. Схема алгоритма**

Алгоритм, разработанный Ривестом, Шамиром и Адлеманом, использует выражения с экспонентами. Данные шифруются блоками, каждый блок рассматривается как число, меньшее некоторого числа n. Шифрование и дешифрование имеют следующий вид для некоторого не зашифрованного блока М и зашифрованного блока С.

С = Ме (mod n)

M = Cd (mod n) = (Me)d (mod n) = Med (mod n)

Как отправитель, так и получатель должны знать значение n. Отправитель знает значение е, получатель знает значение d. Таким образом, *открытый ключ* есть KU = {e, n} и *закрытый ключ* есть KR = {d, n}. При этом должны выполняться следующие условия:

1. Возможность найти *е*, *d* и *n* такие, что *Med = M mod n* для всех М < n .
2. Относительная легкость вычисления Ме и Сd для всех значений М < n.
3. Невозможность определить d, зная е и n.

Рассмотрим некоторые математические понятия, свойства и теоремы, которые позволят нам определить e, d и n.

1. Если (а · b) ≡ (a · c) mod n, то b ≡ c mod n, если а и n взаимнопростые, т.е gcd (a, n) = 1.
2. Обозначим Zp - все числа, взаимнопростые с p и меньшие p. Если p - простое, то Zp - это все остатки. Обозначим w-1 такое число, что w · w- 1 ≡ 1 mod p.

Тогда ∀ w ∈Zp ∃ z: w · z ≡1 mod p

1. Определим функцию Эйлера следующим образом: Φ(n) - число положительных чисел, меньших n и взаимнопростых с n. Если p - простое, то Φ(р) = p-1.

Если p и q - простые, то Φ(p · q) = (p-1) · (q-1).

1. *Теорема Ферма*.

an-1 ≡1 mod n, если n - простое.

1. *Теорема Эйлера*.

aΦ(n) ≡1 mod n для всех взаимнопростых a и n. Теперь рассмотрим все элементы *алгоритма RSA*.

|  |  |
| --- | --- |
| p, q - два простых целых числа | -открыто, вычисляемо. |
| n = p · q | - закрыто, вычисляемо. |
| d, gcd (Φ(n), d) = 1; | - открыто, выбираемо. |
| 1 < d < Φ(n) |
| е ≡ d-1 mod Φ(n) | - закрыты, выбираемы. |

*Закрытый ключ* состоит из {d, n}, *открытый ключ* состоит из {e, n}. Предположим, что пользователь А опубликовал свой *открытый ключ*, и что пользователь В хочет послать пользователю А сообщение М. Тогда В вычисляет С = Ме (mod n) и передает С. При получении этого зашифрованного текста пользователь А дешифрует вычислением М = С d

(mod n).

Суммируем *алгоритм RSA*:

*Создание ключей*

|  |
| --- |
| Выбрать простые р и q |
| Вычислить n = p · q |
| Выбрать d gcd (Φ(n), d) = 1; 1 < d < Φ(n) |
| Вычислить е е = d-1 mod Φ(n) |
| *Открытый ключ* KU = {e, n} |
| *Закрытый ключ* KR = {d, n} |

*Шифрование*

|  |
| --- |
| Незашифрованный текст: М < n |
| Зашифрованный текст: С = М е (mod n) |

*Дешифрование*

|  |
| --- |
| Зашифрованный текст: С |
| Незашифрованный текст: М = Сd (mod n) |

***Обсуждение криптоанализа***

Можно определить четыре возможных подхода для криптоанализа

*алгоритма RSA*:

1. Лобовая атака: перебрать все возможные *закрытые ключи*.
2. Разложить n на два простых сомножителя. Это даст возможность вычислить Φ(n) = (p-1) · (q-1) и d = e-1 (mod Φ(n)).
3. Определить Φ(n) непосредственно, без начального определения р и q. Это также даст возможность определить d = e-1 (mod Φ(n)).
4. Определить d непосредственно, без начального определения Φ(n).

Защита от лобовой атаки для *RSA* и ему подобных алгоритмов приводится далее.

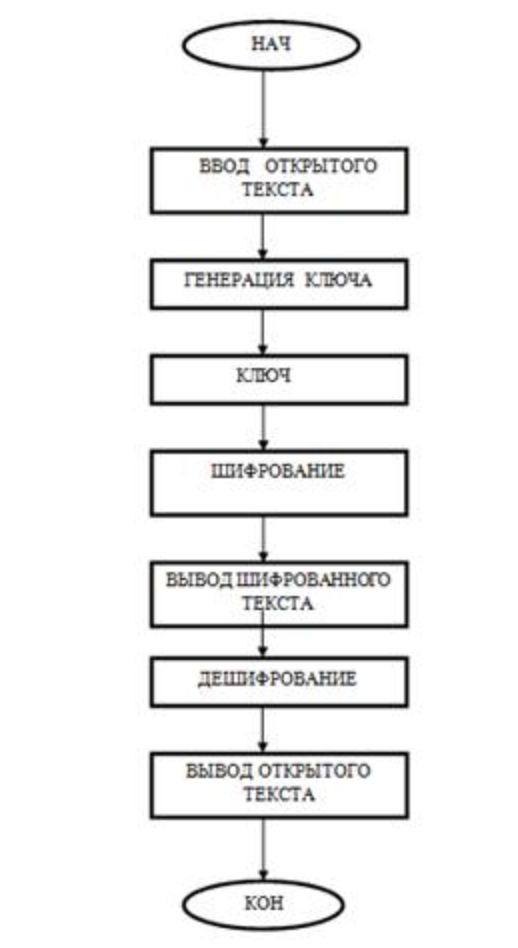


Рисунок 1. Блок-схема алгоритма

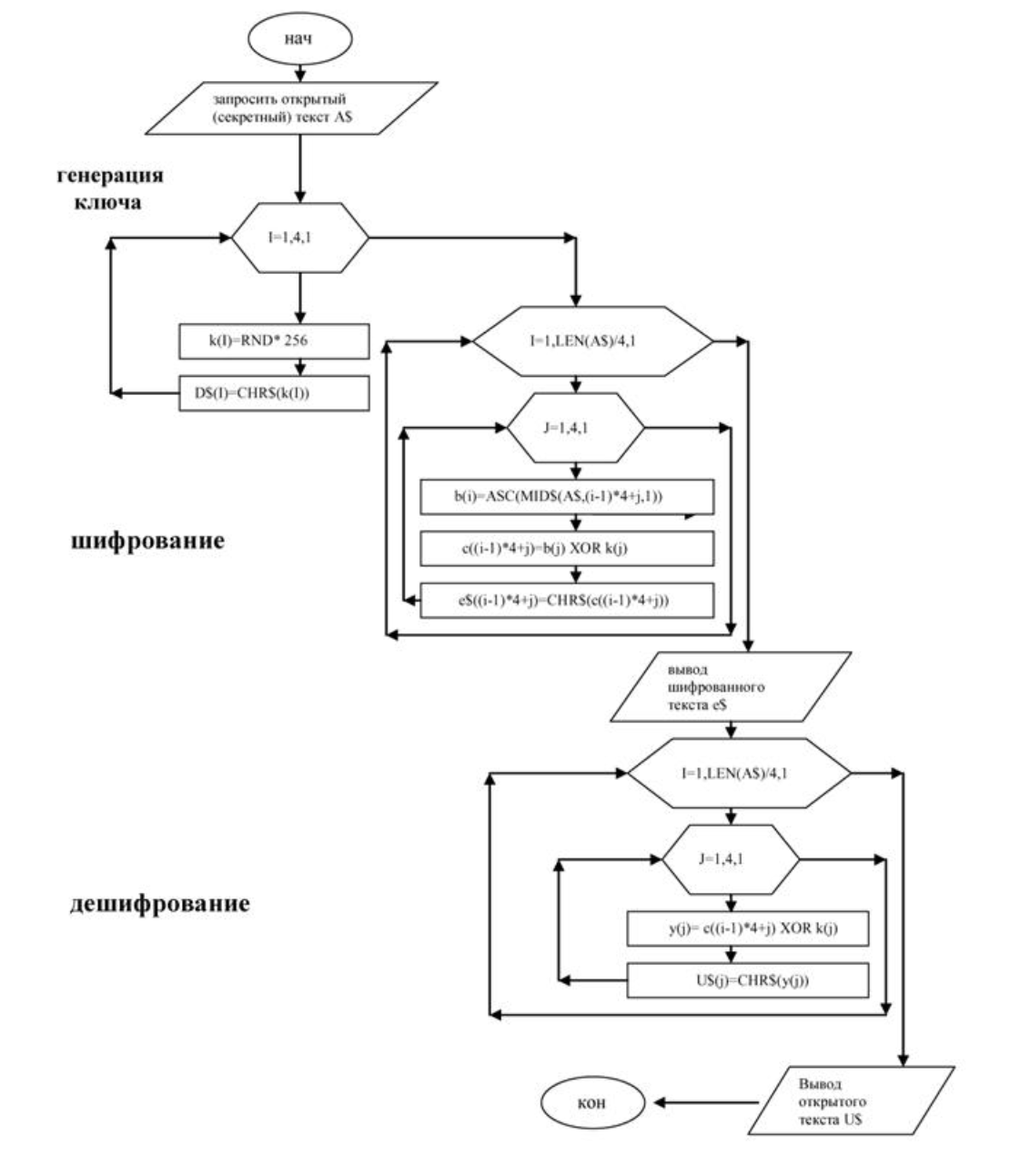


Рисунок 2. Блок-схема полного алгоритма

# 

# 

# **3. Скриншоты ввода данных и результатов выполнения программы**



После начала запуска программа просит ввести сообщение, которое зашифровывается с использованием ключей рандомной выборки, и позднее расшифровывает зашифрованное сообщение.

# **4. Исходные файлы:**

**Source.cpp**

#include<iostream>

#include<math.h>

#include<string.h>

#include<stdlib.h>

#include <cstdlib>

#include <ctime>

using namespace std;

long int p, q, n, t, flag, e[100], d[100], temp[100], j, m[100], en[100], i;

char msg[100];

int prime(long int);

void ce();

long int cd(long int);

void encrypt();

void decrypt();

int prime(long int pr)

{

int i;

j = sqrt(pr);

for (i = 2; i <= j; i++)

{

if (pr % i == 0)

return 0;

}

return 1;

}

int main()

{

srand(time(0));

while (prime(p=rand()%100) == 0){}

while (prime(q = rand() % 100) == 0 || p==q){}

cout << "\nENTER MESSAGE\n";

fflush(stdin);

cin.getline(msg,100);

for (i = 0; msg[i] != '\0'; i++)

m[i] = msg[i];

n = p \* q;

t = (p - 1) \* (q - 1);

ce();

encrypt();

decrypt();

return 0;

}

void ce()

{

int k;

k = 0;

for (i = 2; i < t; i++)

{

if (t % i == 0)

continue;

flag = prime(i);

if (flag == 1 && i != p && i != q)

{

e[k] = i;

flag = cd(e[k]);

if (flag > 0)

{

d[k] = flag;

k++;

}

if (k == 99)

break;

}

}

}

long int cd(long int x)

{

long int k = 1;

while (1)

{

k = k + t;

if (k % x == 0)

return (k / x);

}

}

void encrypt()

{

long int pt, ct, key = e[0], k, len;

i = 0;

len = strlen(msg);

while (i != len)

{

pt = m[i];

pt = pt - 96;

k = 1;

for (j = 0; j < key; j++)

{

k = k \* pt;

k = k % n;

}

temp[i] = k;

ct = k + 96;

en[i] = ct;

i++;

}

en[i] = -1;

cout << "\nTHE ENCRYPTED MESSAGE IS\n";

for (i = 0; en[i] != -1; i++)

printf("%c", en[i]);

}

void decrypt()

{

long int pt, ct, key = d[0], k;

i = 0;

while (en[i] != -1)

{

ct = temp[i];

k = 1;

for (j = 0; j < key; j++)

{

k = k \* ct;

k = k % n;

}

pt = k + 96;

m[i] = pt;

i++;

}

m[i] = -1;

cout << "\nTHE DECRYPTED MESSAGE IS\n";

for (i = 0; m[i] != -1; i++)

printf("%c", m[i]);

}