Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

**ОТЧЕТ О ПРАКТИЧЕСКОЙ** **РАБОТЕ №****2**

«Асимметричные шифры: шифр RSA»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Преподаватель |  |  |  |  | Сидарас А.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ19-09Б, 031940306 |  |  |  | Ману С.А. |
|  | номер группы, зачетной книжки |  | подпись, дата |  | инициалы, фамилия |

Красноярск 2021

**Задание:** реализовать алгоритм ассиметричного шифрования RSAю

**Описание алгоритма:** криптографический алгоритм с открытым ключом, основывающийся на вычислительной сложности задачи факторизации больших целых чисел.

**Код программы:**

import random

from math import gcd

from math import sqrt

from sympy import isprime

def generate\_key\_pair(p, q):

    if not (isprime(p) and isprime(q)):

        raise ValueError('Числа должны быть простыми!')

    elif p == q:

        raise ValueError('Числа не должны быть одинаковыми!')

    n = p \* q

    phi = (p-1) \* (q-1)

    e = random.randrange(1, phi)

    g = gcd(e, phi)

    while g != 1:

        e = random.randrange(1, phi)

        g = gcd(e, phi)

    d = multiplicative\_inverse(e, phi)

    return ((e, n), (d, n))

def multiplicative\_inverse(e, phi):

    d = 0

    x1 = 0

    x2 = 1

    y1 = 1

    temp\_phi = phi

    while e > 0:

        temp1 = temp\_phi // e

        temp2 = temp\_phi - temp1 \* e

        temp\_phi = e

        e = temp2

        x = x2 - temp1 \* x1

        y = d - temp1 \* y1

        x2 = x1

        x1 = x

        d = y1

        y1 = y

    if temp\_phi == 1:

        return d + phi

def encrypt(pk, plaintext):

    key, n = pk

    cipher = [pow(ord(char), key, n) for char in plaintext]

    return cipher

def decrypt(pk, ciphertext):

    key, n = pk

    dba = [str(pow(char, key, n)) for char in ciphertext]

    plain = [chr(int(char2)) for char2 in dba]

    return ''.join(plain)

if \_\_name\_\_ == "\_\_main\_\_":

    print("RSA crypt: ")

    p = int(random.randrange(1, 10000))

    q = int(random.randrange(1, 10000))

    while not isprime(p):

        p = int(random.randrange(1, 10000))

    while not isprime(q):

        q = int(random.randrange(1, 10000))

    public, private = generate\_key\_pair(p, q)

    print("Public key: ", public)

    print("Private key: ", private)

    message = input("Введите сообщение: ")

    encrypted\_msg = encrypt(public, message)

    print("Зашифрованное сообщение: ", ''.join(map(lambda x: str(x) + " ", encrypted\_msg)))

    print("Рассшифрванное сообщение: ", decrypt(private, encrypted\_msg))

**Контрольные примеры:**

Таблица 1 – Контрольные примеры

|  |  |  |  |
| --- | --- | --- | --- |
| **Message** | **Public key** | **Private key** | **Зашифрованное сообщение** |
| Hello world! | (18302483, 25992289) | (28689827, 25992289) | 18362936 6730128 19193294 19193294 2325697 20603353 7435305 2325697 14056814 19193294 12816833 24740396 |
| Siberian Federal University | (4576897, 26145241) | (19095425, 26145241) | 26356070 25766817 14090906 28388528 809724 25766817 31976503 20014355 7780751 25847623 28388528 28063514 28388528 809724 31976503 28397592 7780751 20423131 20014355 25766817 1709018 28388528 809724 11249080 25766817 2364467 14081062 7780751 |
| информационная безопасность | (5604293, 7154299) | (3686477, 7154299) | 3268485 4195970 12176 6438959 83750 4946554 962655 405796 3268485 6438959 4195970 4195970 962655 6381214 2210249 1901106 2975571 6718808 6438959 3472400 962655 4386215 386496 4195970 6438959 4386215 386496 2243948 2210249 |

**Заключение**

Написание данного кода познакомило меня с асимметричными шифрами, в частности RSA.