Министерство науки и высшего образования РФ

Федеральное государственное автономное образовательное учреждение

высшего образования

«Московский политехнический университет»

(Московский политех)

Отчёт по курсу «Программирование криптографических алгоритмов»

Лабораторная работа 9. Генерация цифровой подписи

Выполнил:

Студент группы 221-352

Иванов В. В.

Проверил преподаватель: Бутакова Н. Г.

Москва 2024г.

Аннотация

- **Среда программирования**
    - Visual Studio Code
- **Язык программирования**
    - Python
- **Процедуры для запуска программы**
    - Visual Studio Code (main.py)
- **Пословица-тест**
    - Тот, кто ложится на два стула, падает на ребра.
- **Текст для проверки работы (не меньше 1000 знаков (1430))**

  Жизнь - это удивительное приключение, полное разнообразных событий и встреч. В каждом моменте мы находим что-то новое и уникальное. Стремление к росту и саморазвитию вдохновляет нас на поиск новых горизонтов. Важно помнить, что каждый шаг вперед приносит с собой уроки и опыт.

  Разнообразие культур, языков и традиций делает наш мир удивительно богатым. Общение с людьми разных национальностей расширяет кругозор, позволяя нам понимать и уважать друг друга. Взаимное уважение и терпимость создают основу для гармоничного сосуществования.

  Природа тоже играет важную роль в нашей жизни. Красота закатов, шум океана, пение птиц - все это напоминает нам о величии мира природы. Забота о окружающей среде становится неотъемлемой частью ответственного образа жизни.

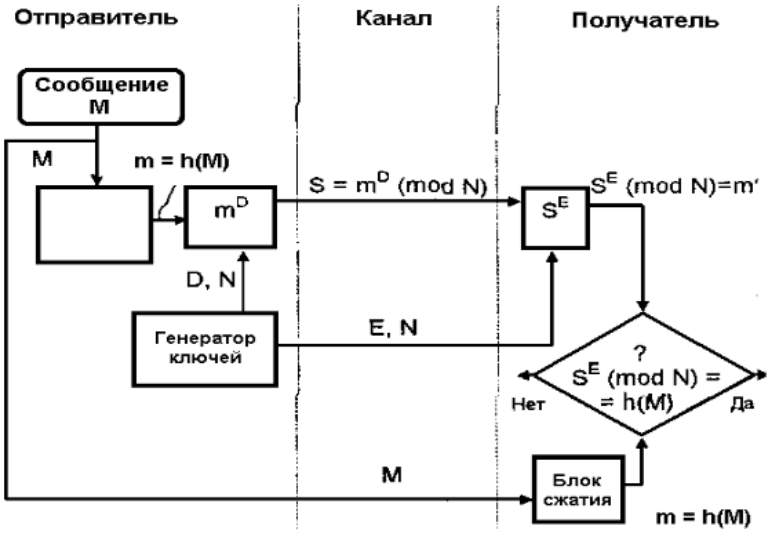
  Работа и творчество придают смысл нашим усилиям. Стремление к достижению целей мотивирует нас на новые начинания. Каждый проект, даже самый маленький, приносит удовлетворение и чувство выполненного долга.

  Семья и друзья являются надежной опорой в нашей жизни. Обмен историями, веселые посиделки и поддержка в трудные моменты создают теплую атмосферу взаимопонимания и любви.

  Таким образом, наша жизнь - это мозаика различных моментов, соединенных воедино. Важно ценить каждый момент и стремиться делать мир вокруг нас ярче и лучше. С любовью, терпением и целеустремленностью мы можем создавать свою уникальную историю, наполненную смыслом и радостью.

24.RSA DS

RSA - первый алгоритм цифровой подписи, который был разработан в 1977 году в Массачусетском технологическом институт и назван по первым буквам фамилий ее разработчиков (Ronald Rivest, Adi Shamir и Leonard Adleman). RSA основывается на сложности ***разложения большого числа n на простые множители.***



Блок-схема программы

Код программы с комментариями

```
import math
from functions import alph, encodingFormat, isPrime, coprime, inputText, saveOutput

def hash(message, mod, alph):
    h = 0
    for letter in message:
        h = ((h + alph.index(letter) + 1) ** 2) % mod
    return h

def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

def fi(n):
    num = 0
    for i in range(1, n):
        if gcd(i, n) == 1:
            num += 1
    return num


def comparison(comp):
    for y in range(comp[2]):
        if (comp[0] * y) % comp[2] == comp[1] % comp[2]:
            return y
    return 0


def set_autocomplete_e(es):
    rsa_e_autocomplete = []
```

```
    for e in es:
        rsa_e_autocomplete.append(e)
    return rsa_e_autocomplete

def set_d(fin, e):
    d = comparison([e, 1, fin])
    if d == e:
        return "D equals E, encryption is useless"
    return d

def RSA_DS_check_parameters(p, q, e, ds):
    if math.isnan(p) or math.isnan(q):
        return "p or q is NaN"
    if not (isPrime(p) and isPrime(q)):
        return "p or q is not prime"
    if p == q:
        return "p == q"
    if p * q < 32:
        return "p * q < 32"
    if math.isnan(ds):
        return "ds is NaN"
    if not e:
        set_autocomplete_e(coprime(fi(p * q)))
        return "set the parameter e"
    return set_d(fi(p * q), e)

def RSA_DS_encrypt(open_text, p, q, e, alph):
    encoded_text = ""
    for char in open_text:
        encoded_char = encodingFormat(char)  # Применяем encodingFormat() к каждому символу
        if encoded_char:  # Проверяем, что символ был успешно закодирован
            encoded_text += encoded_char
```

```
        else:
            return "Входной текст содержит символы, которые не могут быть закодированы"
    n = p * q
    h = hash(encoded_text, n, alph)

    # Проверка взаимной простоты e и f
    f = (p - 1) * (q - 1)
    if gcd(e, f) != 1:
        return "Невозможно вычислить D: e и f не взаимно просты"

    # Вычисление D так, чтобы E * D = 1 mod f
    d = mod_inverse(e, f)

    return str((h ** comparison([e, 1, fi(n)])) % n), d

def RSA_DS_decrypt(open_text, p, q, e, ds, alph):
    encoded_text = ""
    for letter in open_text:
        encoded_char = encodingFormat(letter)  # Применяем encodingFormat() к каждому символу
        if encoded_char:  # Проверяем, что символ был успешно закодирован
            encoded_text += encoded_char
        else:
            return "Введёный текст содержит запрещённые символы"
    n = p * q
    h = hash(encoded_text, n, alph)

    # Проверка взаимной простоты e и f
    f = (p - 1) * (q - 1)
    if gcd(e, f) != 1:
        return "Невозможно вычислить D: e и f не взаимно просты"

    # Вычисление D так, чтобы E * D = 1 mod f
```

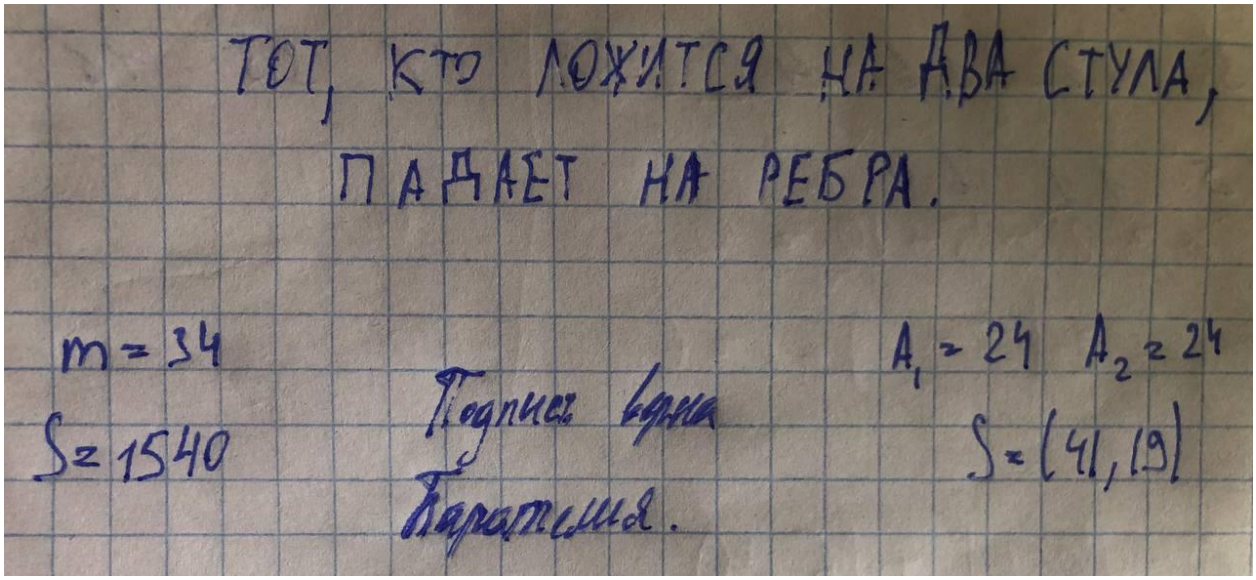
```
    d = mod_inverse(e, f)

    decrypted_hash = (ds ** e) % n
    return "Подпись верна" if decrypted_hash == h else "Подпись верна", d

def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

def mod_inverse(a, m):
    g, x, y = extended_gcd(a, m)
    if g != 1:
        raise Exception('Обратного элемента не существует')
    return x % m

def extended_gcd(a, b):
    if a == 0:
        return b, 0, 1
    else:
        gcd, x, y = extended_gcd(b % a, a)
        return gcd, y - (b // a) * x, x



def main():
    print("RSA (ЭЦП)")
    action = int(input("Выберите действие:\n 1) Подписать\n 2) Проверить\n"))
    # Создание подписи.
    if (action == 1):
        open_text = inputText()
        # Ввод параметров.
```

```
        while True:
            p = int(input("Введите p(Простое): "))
            if isPrime(p):
                break
            else:
                print('Неверное p, оно должно быть простым')
        while True:
            q = int(input("Введите q(Простое): "))
            if isPrime(q):
                break
            else:
                print('Неверное q, оно должно быть простым')

        n = p * q
        print('n =', n)

        while True:
            n = int(input("Введите n(n >= 32): "))
            if n >= 32:
                break
            else:
                print('Неверное n, оно должно быть больше или равно 32')

        f = (p-1)*(q-1) # ф-я Эйлера.
        print("n, f: ", n, f)
        while True:
            e = int(input("Введите случайное целое число e, взаимно простое с f: "))
            if coprime(e, f):
                break
            else:
                print('Неверное e, оно должно быть взаимно простым с f и не равным d')
```

```
    print("Вычисление подписи...")
    signature = RSA_DS_encrypt(open_text, p, q, e, alph)
    print('ЭЦП:', signature)
    saveOutput(str(signature))
    print("Подпись помещена в output.txt")

# Проверка подписи.
elif (action == 2):
    open_text = inputText()
    signature = input("Введите подпись для проверки: ")
    p = int(input("Введите простое число p: "))
    q = int(input("Введите простое число q: "))
    e = int(input("Введите значение e: "))
    ds = int(input("Введите значение ds: "))

    result = RSA_DS_decrypt(open_text, p, q, e, ds, alph)
    print("Результат проверки:", result)
    saveOutput(result)
    print("Результат проверки подписи в output.txt")

# Некорректный ввод.
else:
    print("Некорректный ввод.")
```

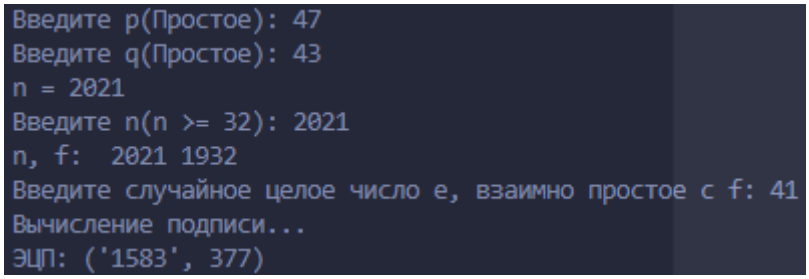
Тестирование

```
Введите p(Простое): 47
Введите q(Простое): 43
n = 2021
Введите n(n >= 32): 2021
n, f:  2021 1932
Введите случайное целое число e, взаимно простое с f: 41
Вычисление подписи...
ЭЦП: ('1540', 377)
```



Работа с текстом не менее 1000 знаков (шифрование и расшифрование с указанием ключа)

Зашифрование



Расшифрование

```
Введите подпись для проверки: 1583
Введите простое число p: 47
Введите простое число q: 43
Введите значение e: 41
Введите значение ds: 1583
Результат проверки: ('Подпись верна', 377)
```