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 4. Обмен ключами по алгоритму Diffie–Hellman

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.

28. Diffie-Hellman

В протоколе обмена секретными ключами предполагается, что все пользователи знают некоторые числа **n** и **a** (1< a < n)*.

Для выработки общего секретного ключа пользователи A и B должны проделать следующую процедуру:

1. Определить секретные ключи пользователей $\mathbf{К_A}$ и $\mathbf{К_B}$.
2. Для этого каждый пользователь независимо выбирает случайные числа из интервала [2, n-1].

3. Вычислить открытые ключи пользователей $\mathbf{Y_A}$ и $\mathbf{Y_B}$:

$$\mathbf{Y=a^K \bmod n}$$

4. Обменяться ключами $\mathbf{Y_A}$ и $\mathbf{Y_B}$ по открытому каналу связи.

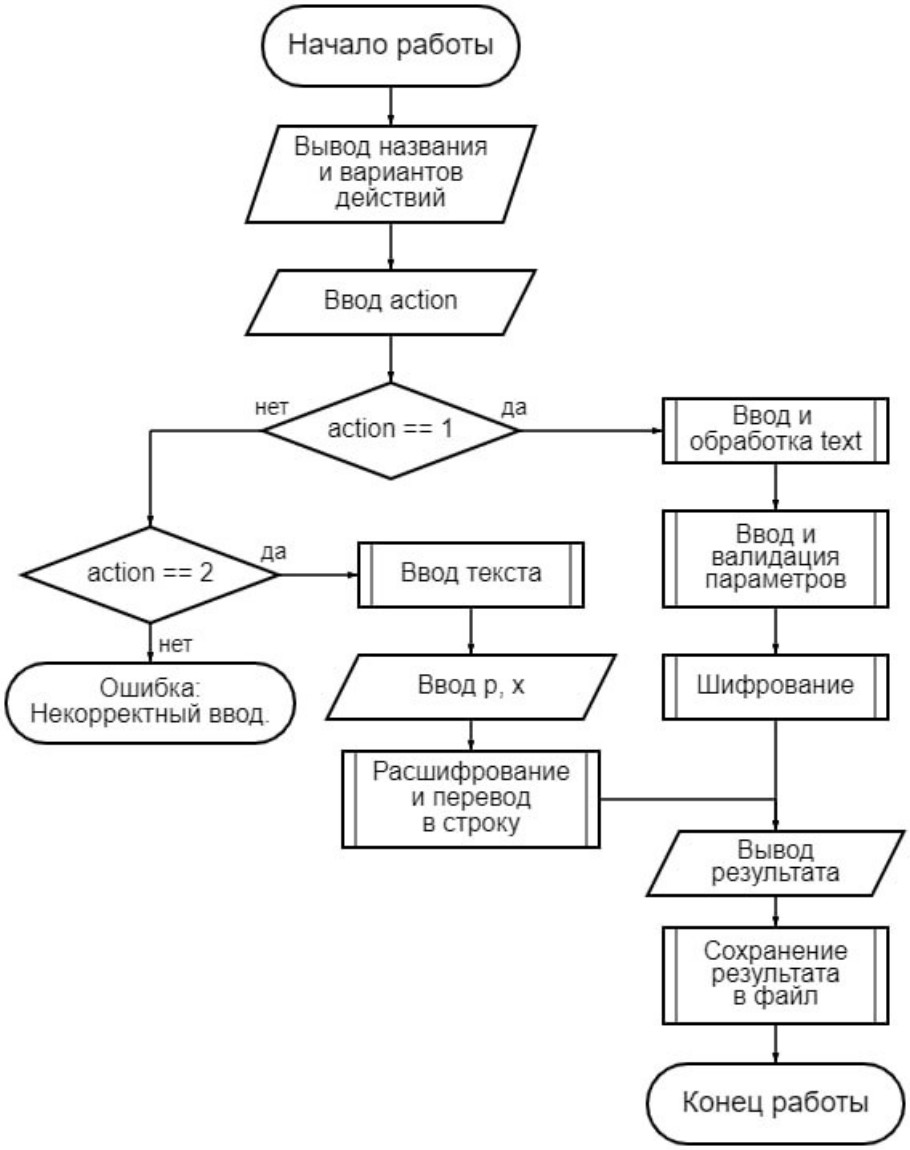
5. Независимо определить общий секретный ключ **К:**

$$\mathbf{K_A=Y^{KA} \bmod n}$$
$$\mathbf{K_B=Y^{KB} \bmod n.}$$

Проверка: *задача решена, если выполняется равенство*

$$\mathbf{K_A = K_B = K}$$

Блок-схема программы



Код программы с комментариями

```
def is_prime(num):
    """Checks if a number is prime."""
    # If the number is less than or equal to 1, it is not prime.
    if num <= 1:
        return False
    # Iterate from 2 to the square root of the number.
    for i in range(2, int(num ** 0.5) + 1):
        # If the number is divisible by any number between 2 and the square root,
        # it is not prime.
        if num % i == 0:
            return False
    # If the number has gone through the loop without finding a divisor,
    # it is prime.
    return True

def set_a_ka_kb_span(n):
    """Sets the text content of the span elements to show the constraints of the Diffie-Hellman key exchange."""
    # Set the text content of the input elements to show the constraints.
    diffie_hellman_a_span = f"a (1 < a < {n})"
    diffie_hellman_ka_span = f"ka (1 < ka < {n})"
    diffie_hellman_kb_span = f"kb (1 < kb < {n})"
    return diffie_hellman_a_span, diffie_hellman_ka_span, diffie_hellman_kb_span

def delete_a_ka_kb_span():
    """Deletes the text content of the span elements to clear the constraints of the Diffie-Hellman key exchange."""
    # Clear the text content of the input elements to delete the constraints.
    diffie_hellman_a_span = "a"
    diffie_hellman_ka_span = "ka"
    diffie_hellman_kb_span = "kb"
```

```
    return diffie_hellman_a_span, diffie_hellman_ka_span, diffie_hellman_kb_span

def diffie_hellman_check_parameters(n, a, ka, kb):
    """Checks if the given parameters are valid for Diffie-Hellman key exchange."""
    # Delete the text content of the input elements and set it back to the default
    delete_a_ka_kb_span()
    # Check if n is not empty
    if not n:
        return "Enter the value of n"
    # Check if n is greater than 2
    if not (2 < n):
        return "n must be greater than 2"
    # Check if n is prime
    if not is_prime(n):
        return "n must be prime"
    # Set the text content of the input elements to show the constraints
    set_a_ka_kb_span(n)
    # Check if a is not empty
    if not a:
        return "Enter the value of a"
    # Check if a is within the constraints
    if not (1 < a < n):
        return "a must be greater than 1 and less than n"
    # Check if ka is not empty
    if not ka:
        return "Enter the value of ka"
    # Check if ka is within the constraints
    if not (1 < ka < n):
        return "ka must be greater than 1 and less than n"
    # Check if kb is not empty
    if not kb:
        return "Enter the value of kb"
```

```
    # Check if kb is within the constraints
    if not (1 < kb < n):
        return "kb must be greater than 1 and less than n"
    # All parameters are valid, return None
    return None

def diffie_hellman(n, a, ka, kb):
    """Calculates Diffie-Hellman key exchange using the provided parameters."""
    # Calculate the public keys and the secret keys using the provided parameters
    ya = pow(a, ka, n)  # Calculate Ya
    yb = pow(a, kb, n)  # Calculate Yb
    xa = pow(yb, ka, n)  # Calculate Xa
    xb = pow(ya, kb, n)  # Calculate Xb
    # Check if Xa and Xb are equal, indicating a valid key exchange
    if xa == xb:
        # If the keys are equal, return a success message with the keys and secret keys
        return f"Открытый ключ Ya = {ya} \nОткрытый ключ Yb = {yb} \nСекретный ключ Xa = {xa} \nСекретный ключ Xb = {xb} \nXa
= Xb ({xa} = {xb}) Подпись верна"
    else:
        # If the keys are not equal, return a failure message
        return "Подпись не верна"
```

Тестирование