

Лабораторная работа № 8

**. Элементы криптографии. Шифрование (кодирование) различных
исходных текстов одним ключом**

Саргсян Арам Грачьевич

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	12
6	Ответы на контрольные вопросы	13
	Список литературы	15

Список иллюстраций

4.1	Название рисунка	11
-----	----------------------------	----

Список таблиц

1 Цель работы

Освоить на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

2 Задание

Два текста кодируются одним ключом (однократное гаммирование). Требуется не зная ключа и не стремясь его определить, прочесть оба текста. Необходимо разработать приложение, позволяющее шифровать и дешифровать тексты P1 и P2 в режиме однократного гаммирования. Приложение должно определить вид шифротекстов C1 и C2 обоих текстов P1 и P2 при известном ключе ; Необходимо определить и выразить аналитически способ, при котором злоумышленник может прочесть оба текста, не зная ключа и не стремясь его определить.

3 Теоретическое введение

Гаммирование представляет собой наложение (снятие) на открытые (зашифрованные) данные последовательности элементов других данных, полученной с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных. Иными словами, наложение гаммы — это сложение её элементов с элементами открытого (закрытого) текста по некоторому фиксированному модулю, значение которого представляет собой известную часть алгоритма шифрования. В соответствии с теорией криптоанализа, если в методе шифрования используется однократная вероятностная гамма (однократное гаммирование) той же длины, что и подлежащий сокрытию текст, то текст нельзя раскрыть. Даже при раскрытии части последовательности гаммы нельзя получить информацию о всём скрываемом тексте. Наложение гаммы по сути представляет собой выполнение операции сложения по модулю 2 (XOR) (обозначаемая знаком \oplus) между элементами гаммы и элементами подлежащего сокрытию текста. Напомним, как работает операция XOR над битами:

$$0 \oplus 0 = 0, 0 \oplus 1 = 1, 1 \oplus 0 = 1, 1 \oplus 1 = 0.$$

Такой метод шифрования является симметричным, так как двойное прибавление одной и той же величины по модулю 2 восстанавливает исходное значение, а шифрование и расшифрование выполняется одной и той же программой[1].

4 Выполнение лабораторной работы

Написал программу, реализующее задание (рис. 4.1).

```
import java.util.ArrayList;
import java.util.List;

public class Main {
    public static void main(String[] args) {
        int a = 'a';
        List<Character> liters = new ArrayList<>();
        for (int i = a; i < a + 32; i++) {
            liters.add((char) i);
        }
        a = '0';
        for (int i = a; i < a + 10; i++) {
            liters.add((char) i);
        }
        a = 'A';
        for (int i = 1040; i < 1072; i++) {
            liters.add((char) i);
        }

        String P1 = "aaaa";
        String P2 = "абвг";
    }
}
```



```

    String key = "cccc";

    hack(P1, P2, liters);
    cipher(P1, key, liters);
}

public static void hack(String P1, String P2, List<Character> liters) {
    List<Character> code = new ArrayList<>();
    for (int i = 0; i < P1.length(); i++) {
        int index1 = liters.indexOf(P1.charAt(i));
        int index2 = liters.indexOf(P2.charAt(i));
        int newIndex = (index1 + index2) % liters.size();
        code.add(liters.get(newIndex));
    }
    System.out.println(code);
    StringBuilder pr = new StringBuilder();
    for (char c : code) {
        pr.append(c);
    }
    System.out.println(pr);
}

public static void cipher(String P1, String key, List<Character> liters) {
    List<Integer> digitsText = new ArrayList<>();
    List<Integer> digitskey = new ArrayList<>();

    for (char c : P1.toCharArray()) {
        digitsText.add(liters.indexOf(c));
    }
}

```

```

System.out.println("Числовой вариант текста " + digitsText);

for (char c : key.toCharArray()) {
    digitskey.add(liters.indexOf(c));
}
System.out.println("Числовой вариант ключа " + digitskey);

List<Integer> digitsResult = new ArrayList<>();
int ch = 0;
for (char c : P1.toCharArray()) {
    int a = (digitsText.get(ch) + digitskey.get(ch)) % liters.size();
    digitsResult.add(a);
    ch++;
}
System.out.println("Числовой вариант шифротекста " + digitsResult);

StringBuilder textCr = new StringBuilder();
for (int i : digitsResult) {
    textCr.append(liters.get(i));
}
System.out.println("Шифротекст " + textCr);

List<Integer> digits = new ArrayList<>();
for (char c : textCr.toString().toCharArray()) {
    digits.add(liters.indexOf(c));
}
ch = 0;
List<Integer> digits1 = new ArrayList<>();
for (int i : digits) {

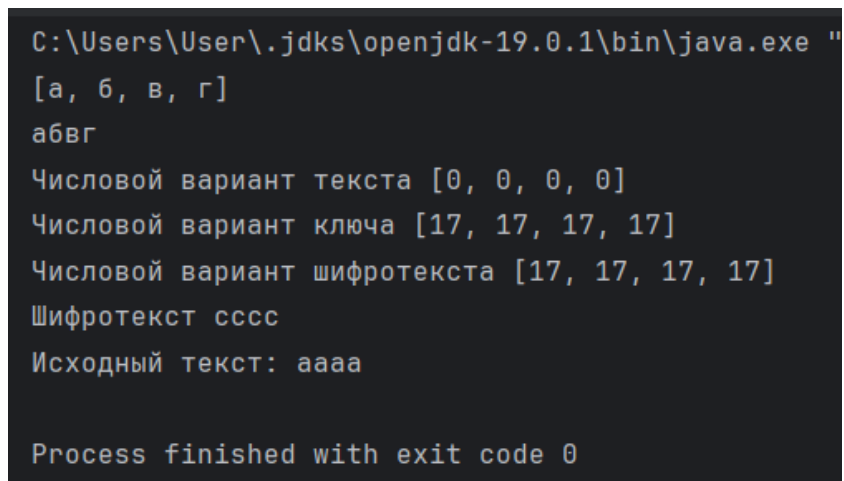
```

```

        int a = (i - digitskey.get(ch) + liters.size()) % liters.size();
        digits1.add(a);
        ch++;
    }

    StringBuilder textDecr = new StringBuilder();
    for (int i : digits1) {
        textDecr.append(liters.get(i));
    }
    System.out.println("Исходный текст: " + textDecr);
}
}

```



```

C:\Users\User\.jdk\openjdk-19.0.1\bin\java.exe "
[a, б, в, г]
абвг
Числовой вариант текста [0, 0, 0, 0]
Числовой вариант ключа [17, 17, 17, 17]
Числовой вариант шифротекста [17, 17, 17, 17]
Шифротекст сссс
Исходный текст: аааа

Process finished with exit code 0

```

Рис. 4.1: Название рисунка

5 Выводы

Я освоил на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

6 Ответы на контрольные вопросы

Однократное гаммирование (также известное как шифр Вернама или шифр по модулю 2) является методом шифрования, который использует одноразовый ключ, который должен быть такой же длины, что и шифруемый текст. Ответим на ваши вопросы:

1. Как, зная один из текстов (P1 или P2), определить другой, не зная при этом ключа?

Если вы знаете один из открытых текстов (P1 или P2) и соответствующий ему зашифрованный текст (C1 или C2), и у вас нет ключа, вы не сможете определить другой открытый текст. Однократное гаммирование обеспечивает сильную криптографическую защиту, если ключ используется только один раз.

2. Что будет при повторном использовании ключа при шифровании текста?

Повторное использование ключа при однократном гаммировании полностью компрометирует шифрование. Если ключ используется повторно, злоумышленный анализатор может просто выполнить операцию XOR между зашифрованным текстом и известным текстом, чтобы получить оригинальный ключ, и затем использовать этот ключ для расшифровки других сообщений, зашифрованных с тем же ключом.

3. Как реализуется режим шифрования однократного гаммирования одним ключом двух открытых текстов?

В однократном гаммировании каждому биту открытого текста соответствует бит ключа. Операция XOR выполняется между открытым текстом и ключом, чтобы получить зашифрованный текст. Если вы хотите зашифровать два разных открытых текста (P1 и P2) с использованием одного и того же ключа, вы просто выполняете операцию XOR между каждым открытым текстом и ключом по отдельности.

4. Перечислите недостатки шифрования одним ключом двух открытых текстов.

Основными недостатками шифрования одним ключом двух открытых текстов (повторное использование ключа) являются: - Уязвимость к криптоанализу при многократном использовании ключа. - Невозможность безопасного обмена ключами для секретной передачи. - Ограничение на длину ключа, который должен быть такой же длины, что и открытый текст.

5. Перечислите преимущества шифрования одним ключом двух открытых текстов.

Преимущества такого шифрования ограничены и обычно не рекомендуются. Однако можно выделить следующие преимущества: Простота реализации: Однократное гаммирование легко реализовать с использованием операции XOR. Высокая скорость шифрования и дешифрования: Поскольку операция XOR быстрая, шифрование и дешифрование могут выполняться быстро.

Список литературы

1. Abdullaev T.R., Juraev G.U. Application three-valued logic in symmetric block encryption algorithms // Journal of Physics: Conference Series. IOP Publishing, 2021. T. 2131, № 2. С. 022082.