

# **Отчёт по лабораторной работе №7**

**Шифр гаммирования**

Радойичич Милица НФИбд-03-18

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Теоретические сведения</b>	<b>5</b>
2.1	Шифр гаммирования . . . . .	5
<b>3</b>	<b>Выполнение работы</b>	<b>7</b>
3.1	Реализация шифратора и дешифратора на Java . . . . .	7
3.2	Контрольный пример . . . . .	13
<b>4</b>	<b>Выводы</b>	<b>14</b>
	<b>Список литературы</b>	<b>15</b>

# List of Figures

3.1	Работа алгоритма гаммирования . . . . .	13
-----	---	----

# 1 Цель работы

Изучение алгоритма шифрования гаммированием

## 2 Теоретические сведения

### 2.1 Шифр гаммирования

Гаммирование – это наложение (снятие) на открытые (зашифрованные) данные криптографической гаммы, т.е. последовательности элементов данных, вырабатываемых с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных.

Принцип шифрования гаммированием заключается в генерации гаммы шифра с помощью датчика псевдослучайных чисел и наложении полученной гаммы шифра на открытые данные обратимым образом (например, используя операцию сложения по модулю 2). Процесс дешифрования сводится к повторной генерации гаммы шифра при известном ключе и наложении такой же гаммы на зашифрованные данные. Полученный зашифрованный текст является достаточно трудным для раскрытия в том случае, если гамма шифра не содержит повторяющихся битовых последовательностей и изменяется случайным образом для каждого шифруемого слова. Если период гаммы превышает длину всего зашифрованного текста и неизвестна никакая часть исходного текста, то шифр можно раскрыть только прямым перебором (подбором ключа). В этом случае криптостойкость определяется размером ключа.

Метод гаммирования становится бессильным, если известен фрагмент исходного текста и соответствующая ему шифрограмма. В этом случае простым вычитанием по модулю 2 получается отрезок псевдослучайной последовательности и по нему восстанавливается вся эта последовательность.

Метод гаммирования с обратной связью заключается в том, что для получения сегмента гаммы используется контрольная сумма определенного участка шифруемых данных. Например, если рассматривать гамму шифра как объединение непересекающихся множеств  $H(j)$ , то процесс шифрования можно представить следующими шагами:

1. Генерация сегмента гаммы  $H(1)$  и наложение его на соответствующий участок шифруемых данных.
2. Подсчет контрольной суммы участка, соответствующего сегменту гаммы  $H(1)$ .
3. Генерация с учетом контрольной суммы уже зашифрованного участка данных следующего сегмента гаммы  $H(2)$ .
4. Подсчет контрольной суммы участка данных, соответствующего сегменту данных  $H(2)$  и т.д.

## 3 Выполнение работы

### 3.1 Реализация шифратора и дешифратора на Java

```
import java.util.HashMap;  
import java.util.Iterator;  
import java.util.Map;  
import java.util.Scanner;
```

```
public class Shifrovka {  
    public static void main(String [] args) {
```

```
        HashMap<Character, String> map = new HashMap<Character ,String>();  
        map.put('0', "0000");  
        map.put('1', "0001");  
        map.put('2', "0010");  
        map.put('3', "0011");  
        map.put('4', "0100");  
        map.put('5', "0101");  
        map.put('6', "0110");  
        map.put('7', "0111");  
        map.put('8', "1000");
```

```

map.put('9', "1001");
map.put('A', "1010");
map.put('B', "1011" );
map.put('C', "1100");
map.put('D', "1101");
map.put('E', "1110" );
map.put('F', "1111");


String text="";
String cipher;
String cipher2;


Scanner in = new Scanner(System.in);
System.out.println("введите '1' если хотите определить шифротекст по ключу и
int input = in.nextInt();
if(input==1) {
    Scanner in2 = new Scanner(System.in);
    System.out.println("введите ключ шифрования (ключ должен быть в шестнадцат
cipher= in2.nextLine();
    System.out.println("введите открытый текст (размерность текста должна совпа
cipher2 = in2.nextLine();
    cipher2= characterto16(cipher2,map);
}else {
    Scanner in2 = new Scanner(System.in);
    System.out.println("введите шифротекст : ");
    cipher= in2.nextLine();


    System.out.println("введите открытый текст(размерность текста должна совпа

```



```

        cipher2= in2.nextLine();
        cipher2= characterto16(cipher2,map);
    }

    String shifr = shifrovanie(cipher,cipher2,map);

    if(input==1) {
        System.out.println("шифротекст : "+shifr);
    }else {
        System.out.println("ключ : "+shifr);
    }

}

public static String characterto16 (String cipher,HashMap<Character, String> map)
    char[] chararray = cipher.toCharArray();
    String finalcode="";
    for(int i=0;i<chararray.length;i++) {
        char character = chararray[i];
        int ascii = (int) character;
        String code = Integer.toString(ascii,2);
        String curcode=code;
        for(int j=0;j<8-code.length();j++) {
            curcode="0"+curcode;
        }
        code= curcode;
        String val = code.substring(0, 4);

```

```

String val2= code.substring(4);
char nval=' ';
char nval2=' ';
Iterator it = map.entrySet().iterator();

while (it.hasNext()) {
    Map.Entry pair = (Map.Entry)it.next();
    if(pair.getValue().equals(val)) {
        nval=(char)pair.getKey();
    }

    if(pair.getValue().equals(val2)) {
        nval2=(char)pair.getKey();
    }

}

String v = String.valueOf(nval)+String.valueOf(nval2);
finalcode=finalcode+v+" ";

}

return finalcode;

}

public static String shifrovanie(String cipher, String cipher2,HashMap<Character,

String[] splt = cipher.split("\\s+");

```

```

String[] splt2 = cipher2.split("\\s+");

String finalcode="";
for(int i=0;i<splt.length;i++) {

char[] symbols = splt[i].toCharArray();
String symbol = map.get(symbols[0])+map.get(symbols[1]);

char[] symbols2 = splt2[i].toCharArray();
String symbol2 = map.get(symbols2[0])+map.get(symbols2[1]);

String newsymbol="";
for(int j=0;j<symbol2.length();j++) {

int number= Character.digit(symbol2.charAt(j), 10);
int number2 = Character.digit(symbol.charAt(j), 10);

newsymbol+=number^number2;

}

String val = newsymbol.substring(0, 4);
String val2= newsymbol.substring(4);
char nval=' ';
char nval2=' ';
Iterator it = map.entrySet().iterator();

while (it.hasNext()) {

```

```

        Map.Entry pair = (Map.Entry)it.next();
        if(pair.getValue().equals(val)) {
            nval=(char)pair.getKey();
        }

        if(pair.getValue().equals(val2)) {
            nval2=(char)pair.getKey();
        }

    }

    String v = String.valueOf(nval)+String.valueOf(nval2);
    finalcode=finalcode+v+" ";

}

return finalcode;
}

}

```

## 3.2 Контрольный пример

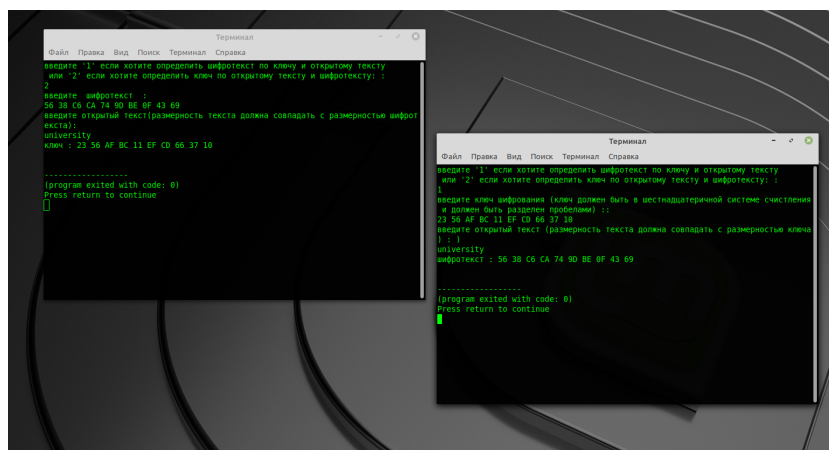


Figure 3.1: Работа алгоритма гаммирования

## **4 Выводы**

Изучили алгоритмы шифрования на основе гаммирования

# Список литературы

1. Шифрование методом гаммирования
2. Режим гаммирования в блочном алгоритме шифрования