МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Лабораторная работа № 4**

**ИССЛЕДОВАНИЕ КРИПТОГРАФИЧЕСКИХ**

**ШИФРОВ НА ОСНОВЕ ПОДСТАНОВКИ**

**(ЗАМЕНЫ) СИМВОЛОВ**

Разработал: Бай И.О.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М.Г.

Минск 2023

**Цель:** изучение и приобретение практических навыков разработки и использования приложений для реализации подстановочных шифров.

**Задачи**:

1. Закрепить теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости подстановочных шифров.
2. Ознакомиться с особенностями реализации и свойствами различных подстановочных шифров на основе готового программного средства (L\_LUX).
3. Разработать приложение для реализации указанных преподавателем методов подстановочного зашифрования/расшифрования.
4. Выполнить исследование криптостойкости шифров на основе статистических данных о частотах появления символов в исходном и зашифрованном сообщениях.
5. Оценить скорость зашифрования/расшифрования реализованных способов шифров.
6. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

**Практическое задание**

|  |  |  |
| --- | --- | --- |
| Вариант | Алфавит | Шифр |
| 2 | Русский | 1. На основе аффинной системы подстановок Цезаря;   *a* = 7, *b* = 10   1. Виженера, ключевое слово – собственная фамилия |

***Представление русского алфавита:***

******

Рис. 1 – Представление русского алфавита

1. ***Аффинная система подстановок Цезаря.***

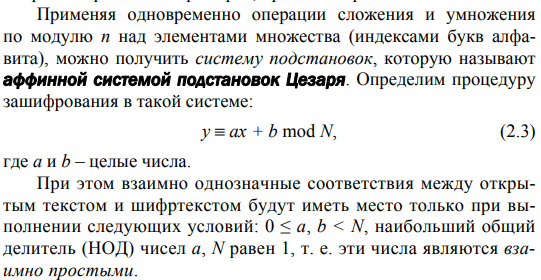
******

Рис. 2 – Алгоритм действий (2.3)

Для реализации аффинной системы подстановок Цезаря (2.3) был разработан класс *CryprtoMethods*. Листинг данного класса представлен ниже.

public static void CesarEncrypt(TextArea text, TextArea result, int a, int b, Label time) {  
 long start = System.*currentTimeMillis*();  
 String res = "";  
 for (int i = 0; i < text.getText().length(); i++) {  
 if (*alphabet*.indexOf(text.getText().charAt(i)) == -1) {  
 res += text.getText().charAt(i);  
 } else {  
 int alphaIndex = *alphabet*.indexOf(text.getText().charAt(i));  
 int troublesome = (a \* alphaIndex + b) % *alphabet*.length();  
 res += *alphabet*.charAt(troublesome);  
 }  
 }  
 long finish = System.*currentTimeMillis*();  
 long elapsed = finish - start;  
 time.setText(elapsed + "мс");  
 result.setText(res);  
 *MakeRecord*(0,1,2, "CesarEncrypt", res);  
}  
  
public static void CesarDecrypt(TextArea text, TextArea result, int a, int b, Label time) {  
 long start = System.*currentTimeMillis*();  
 String res = "";  
 a %= *alphabet*.length();  
 int invert = 0; //обратное к a число по модулю N  
 for (int j = 1; j < *alphabet*.length(); j++) {  
 if ((a \* j) % *alphabet*.length() == 1) {  
 invert = j;  
 }  
 }  
  
 for (int i = 0; i < text.getText().length(); i++) {  
 if (*alphabet*.indexOf(text.getText().charAt(i)) == -1) {  
 res += text.getText().charAt(i);  
 } else {  
 int alphaIndex = *alphabet*.indexOf(text.getText().charAt(i));  
 int troublesome = (invert \* (alphaIndex - b)) % *alphabet*.length();  
 if (troublesome < 0) {  
 troublesome += *alphabet*.length();  
 }  
 res += *alphabet*.charAt(troublesome);  
 }  
 }  
 long finish = System.*currentTimeMillis*();  
 long elapsed = finish - start;  
 time.setText(elapsed + "мс");  
 result.setText(res);  
 *MakeRecord*(4,5,6, "CesarDecrypt", res);  
}

Листинг 1 – *CryptoMethods.java*

Реализация аффинной системы подстановок Цезаря, *a* = 7, *b* = 10:

Табл. 1 – Результаты выполнения аффинной системы подстановок Цезаря.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исходный текст | Зашифрованный текст | Время шифрования | Расшифрованный текст | Время расшифрования |
| Устная передача — самый древний способ передачи знаний в истории человечества. | Удкийг цлэлей-мй — дйвзн еэлчижн дцпдпр цлэлеймж аий-ижн ч ждкпэжж млыпчлмлдкчй. | 6 мс | Устная переда-ча — самый древний способ передачи зна-ний в истории человечества. | 1 мс |

Гистограммы появления символов:

Рис. 3 – Гистограммы появления символов CesarDecrypt

Рис. 4 – Гистограммы появления символов CesarEncrypt

1. ***Виженер.***

В этом шифре мы имеем дело с последовательностью сдвигов, циклически повторяющейся. Основная идея заключается в следующем. Создается таблица (таблица Виженера) размером N×N (N – число знаков в используемом алфавите). Эти знаки могут включать не только буквы, но и, например, пробел или иные знаки. В первой строке таблицы записывается весь используемый алфавит. Каждая последующая строка получается из предыдущего циклического сдвига последней на 1 символ влево. Таким образом, при мощности алфавита (английского языка), равной 26, необходимо выполнить последовательно 25 сдвигов для формирования всей таблицы.

Весь функционал для шифрования методом Виженера представлен в классе *CryptoMethods*. Листинг данного класса:

public static void VigenereEncrypt(TextArea text, TextArea result, String key, Label time) {  
 long start = System.*currentTimeMillis*();  
 int[] keyValue = new int[key.length()];  
  
 for (int i = 0; i < key.length(); i++) {  
 keyValue[i] = *alphabet*.indexOf(key.charAt(i));  
 }  
  
 String res = "";  
 int pointer = 0;  
  
 for (int i = 0; i < text.getText().length(); i++) {  
 if (*alphabet*.indexOf(text.getText().charAt(i)) != -1) {  
 int shift = (*alphabet*.indexOf(text.getText().charAt(i)) + keyValue[pointer]) % *alphabet*.length();  
 res += *alphabet*.charAt(shift);  
 pointer += 1;  
 } else {  
 pointer = 0;  
 res += text.getText().charAt(i);  
 }  
  
 pointer = pointer >= keyValue.length ? 0 : pointer;  
 }  
 long finish = System.*currentTimeMillis*();  
 long elapsed = finish - start;  
 time.setText(elapsed + "мс");  
 result.setText(res);  
 *MakeRecord*(8,9,10, "VigenereEncrypt", res);  
}  
  
public static void VigenereDecrypt(TextArea text, TextArea result, String key, Label time) {  
 long start = System.*currentTimeMillis*();  
 int[] keyValue = new int[key.length()];  
  
 for (int i = 0; i < key.length(); i++) {  
 keyValue[i] = *alphabet*.indexOf(key.charAt(i));  
 }  
  
 String res = "";  
 int pointer = 0;  
  
 for (int i = 0; i < text.getText().length(); i++) {  
 if (*alphabet*.indexOf(text.getText().charAt(i)) != -1) {  
 int shift = *alphabet*.indexOf(text.getText().charAt(i)) - keyValue[pointer];  
 if (shift < 0) {  
 res += *alphabet*.charAt(*alphabet*.length() - Math.*abs*(shift));  
 } else {  
 res += *alphabet*.charAt(Math.*abs*(shift));  
 }  
 pointer += 1;  
 } else {  
 pointer = 0;  
 res += text.getText().charAt(i);  
 }  
  
 pointer = pointer >= keyValue.length ? 0 : pointer;  
 }  
 long finish = System.*currentTimeMillis*();  
 long elapsed = finish - start;  
 time.setText(elapsed + "мс");  
 result.setText(res);  
 *MakeRecord*(12,13,14, "VigenereDecrypt", res);  
}

Листинг 2 – *CryptoMethods.java*

Табл. 2 – Результаты выполнения. Шифр Виженера

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исходный текст | Зашифрованный текст | Время шифрования | Расшифрованный текст | Время расшифрования |
| Устная передача — самый древний способ передачи знаний в истории человечества. | Уъхьры шзяхаи-аг — ъгылё муутйсм ътэвкй шзяхаиъч рро-цлш к сфбямсл азъяюнъувокг. | 3 мс | Устная передача — самый древний способ передачи знаний в истории человечества. | 4 мс |

Гистограммы появления символов:

Рис. 5 – Гистограммы появления символов VigenerDecrypt

Рис. 6 – Гистограммы появления символов VigenerEncrypt

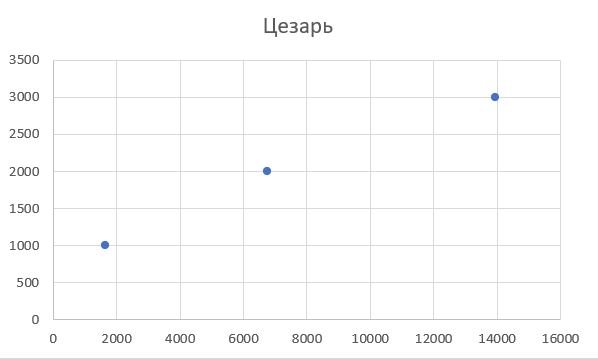


Рис. 7 – Временные зависимости для шифрования Цезарем

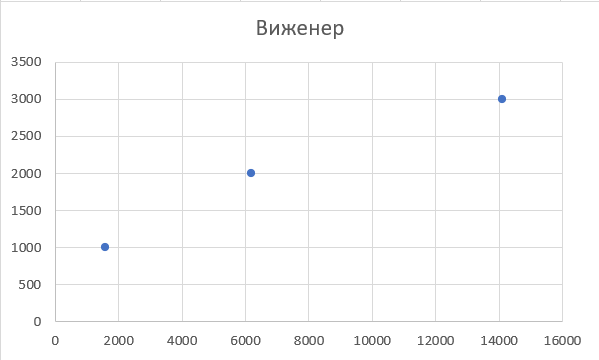


Рис. 7 – Временные зависимости для шифрования Виженером

**Вывод:** в данной лабораторной работе были закреплены теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости шифров. Были ознакомлены с особенностями реализации и свойствами различных подстановочных шифров. Было разработано приложение для реализации указанных преподавателем методов подстановочного зашифрования/расшифрования. Были проведены исследования криптостойкости шифров на основе статистических данных о частотах появления символов в исходном и зашифрованном сообщениях.