1. Цель работы

Реализовать атаку по известному шифртексту при помощи частотного анализа на полиалфавитный шифр подстановки.

2. Описание алгоритма

2.1. Шифр Виженера

Шифр Виженера — метод полиалфавитного шифрования буквенного текста с использованием ключевого слова.

В шифре Цезаря каждая буква алфавита сдвигается на несколько позиций в зависимости от ключа. Так, например, при сдвиге на 4, A стало бы E.

Шифр Виженера состоит из нескольких шифров Цезаря с различными значениями сдвига. Для шифрования может использоваться таблица алфавитов, называемая квадрат (или таблица) Виженера. Например, для английского алфавита таблица Виженера составляется из строк по 26 символов (рис. 1), первой строчкой которой является алфавит, а каждая следующая строка получается из предыдущей путем циклического сдвига вправо на одну позицию. Таким образом получается 26 различных шифров Цезаря.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Рис. 1 Таблица Виженера для английского алфавита

Человек, посылающий сообщение, циклически записывает ключевое слово до тех пор, пока его длина не будет соответствовать длине исходного текста.

Пример:

Ключевое слово:	LEMON
Исходный текст:	ATTACKATDAWN
Циклический ключ:	LEMONLEMONLE
Зашифрованный текст:	LXFOPVEFRBHR

Первый символ исходного текста A зашифрован последовательностью L, которая является первым символом ключа. Первый символ L зашифрованного текста находится на пересечении строки L и столбца A в таблице Виженера (рис. 1). Аналогично для остальных символов.

Расшифровывание производится следующим образом: находим в таблице Виженера строку, соответствующую i-му символу ключевого слова, в данной строке находим i-ый символ зашифрованного текста. Столбец, в котором находится данный символ, соответствует i-му символу исходного текста.

Таким образом, если алфавиту сопоставить числа 0-N, то шифрование можно записать в виде формулы:

$$C_i \equiv (M_i + K_i) mod N \tag{2.1}$$

Тогда расшифровывание представляется в виде:

$$M_i \equiv (C_i - K_i + N) mod N \tag{2.2}$$

2.2. Индекс совпадений

Основной задачей для взлома шифра Виженера является поиск длины ключа. Индекс совпадений — один из методов криптоанализа шифра Виженера.

Рассмотрим текст, написанный на некотором языке, алфавит которого состоит из N букв. Рассмотрим достаточно длинную последовательность $\vec{x}$из nбукв. Если f_iзадает количество i-ой буквы алфавита в строке $\vec{x}$, то можно определить индекс совпадений:

$$I(\vec{x}) = \sum_i \quad f_i \frac{f_i - 1}{n(n-1)} \tag{2.3}$$

Так, если текст написан на естественном языке, является достаточно длинным, то для него можно получить ожидаемый индекс совпадений (например, для английского алфавита индекс совпадения равен примерно 0,065).

Для того, чтобы определить длину ключа, необходимо последовательно записывать в отдельную строку каждую k-ую (k = 2, 3, 4, …) букву и посчитать индекс совпадения для этой строки. Число k, для которого индекс совпадения имеет пиковый характер, будет являться длиной ключа.

2.3. Частотный анализ

Как только длина ключа становится известной, зашифрованный текст можно записать во множество столбцов, каждый из которых соответствует одному символу ключа. Каждый столбец состоит из исходного текста, который зашифрован шифром Цезаря.

Самым простым способом, чтобы определить, каким ключом был зашифрован каждый столбец, является (не всегда такой способ будет успешным):

- посчитать частоту встречаемости каждого символа алфавита в открытом тексте;
- посчитать частоту встречаемости каждого символа алфавита в шифртексте;
- сопоставить максимумы — разность их индексов и будет ключом.

3. Описание реализации

3.1. Класс Vizhener

Является основным классом. Имеет методы для шифрования, дешифрования, генерации произвольного ключа. Также присутствует метод для совершения атаки на зашифрованный текст шифром Виженера. Размер используемого алфавита — 256 символов (таблица ASCII).

`public static void encoder(File text, File encoded, String key);`

Принимает на вход файл с открытым текстом, файл, куда записать зашифрованный текст, и ключ. Реализована формула (2.1).

```
for (int i = 0; i < buffer.length(); i++) {
      buffer[i] = (buffer[i] + key.charAt(i % key.length())
% ASCII_TABLE_SIZE;
}
```

`public static void decoder(File encoded, File decoded, String key);`

Принимает на вход файл с зашифрованным текстом, файл, куда записать расшифрованный текст, и ключ. Реализована формула (2.2).

```
for (int i = 0; i < buffer.length(); i++) {
      buffer[i] = (buffer[i] - key.charAt(i % key.length()
+ ASCII_TABLE_SIZE) % ASCII_TABLE_SIZE;
}
```

public static String keyGen(int keyLength);

Генерирует и возвращает ключ длиной, указанной в качестве параметра.

public static String doAttack(File statistics, File encoded);

Метод совершает атаку на зашифрованный файл. В параметрах указывается файл с открытым текстом для статистики и файл с зашифрованным текстом. Здесь запускается метод **int keyLength = IndexMetod.run(encoded)**. Далее собирается статистика для открытого текста достаточно большого объема методом **int[] statisticsArray = LetterFrequency.getStatistics(statistics).** Полученные статистика открытого текста и возможный ключ, а также зашифрованный файл передаются методу **String key = KeyAnalysis.run(statisticsArray, encoded, keyLength)**, который после завершения вернет ключ.

3.2. Класс IndexMetod

Класс предназначен для поиска длины ключа, которым было зашифровано сообщение, с помощью метода индексов.

public static int run(File text);

Метод считывает текст из файла в буффер. Далее составляет последовательность $\vec{x}$длиной $n = \frac{N}{i}, N -$ размербуффера, добавляя в нее каждую i-ую букву текста для $2 \leq i \leq K_{max}$, где K_{max}— максимальная длина ключа. Считает частоты появления каждого символа в последовательности методом `toCount` класса `LetterFrequency`, а затем вычисляет индекс совпадения по формуле (2.3). Возвращаемым значением будет являться первое пиковое значение. Достигается это следующим способом:

```
if (Math.abs(currentIndex - maxIndex) > INDEX_FLAG) {
      maxIndex = index;
      keyLength = i;
}
```

`INDEX_FLAG` – значение, незначительно меньше индекса совпадения для открытого текста того же алфавита (для английского алфавита примерно равен 0,065). Таким образом

будет выбран первое пиковое значение, т. к. между собой пиковые значения отличаются незначительно.

3.3. Класс KeyAnalisis

Класс предназначен для поиска ключа.

```
public static String run(int[] statistics, File encoded, int keyLength);
```

Выполняет поиск ключа. Создает keyLenght массивов, в каждый из которых добавляет i-ый по модулю keyLength символ зашифрованного текста. Теперь в каждом из этих массивов получается зашифрованный шифром Цезаря текст. Считает частоты встречаемости символов в каждом из этих массивов отдельно. Сортирует частоты по убыванию. Массив со статистикой открытого текста также отсортирован. Далее считающая разности номеров между каждыми i-ыми элементами в массивах частот. Самая часто встречаемая разность определяется как элемент ключа.

3.4. Класс LetterFrequency

Класс предназначен для подсчета частот символов.

```
public static int[] toCount(char[] x);
```

Получает на вход последовательность символов, возвращает массив с частотой встречаемости каждого символа.

```
public static int[] getStatistics(File file)
```

Получает на вход файл с текстом, возвращает массив с частотой встречаемости каждого символа.

4. Примеры работы программы

Сгенерируем случайный ключ длиной 50:
yW2ngnA?xOjyi?vCQNpSZkrT0^IvWhS6o@9LRIvOXXH@iUTd=3

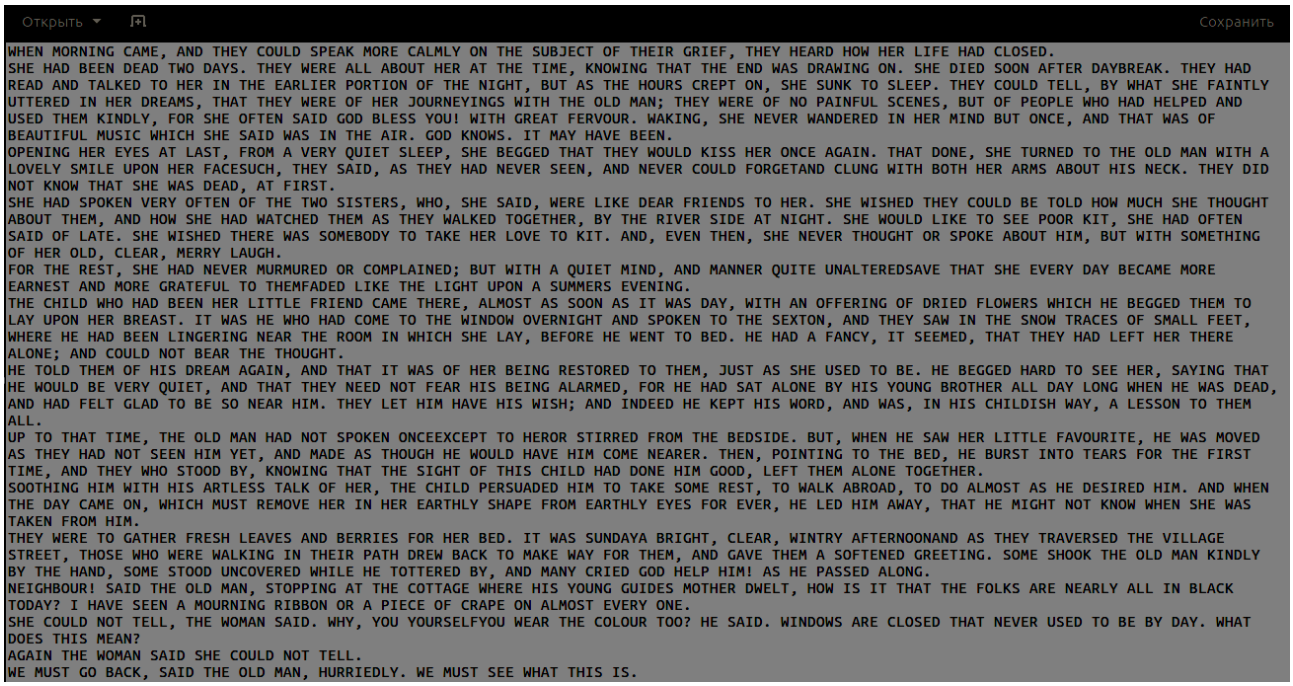

Рис.1 Сообщение для шифрования

Рис.2 Зашифрованное сообщение

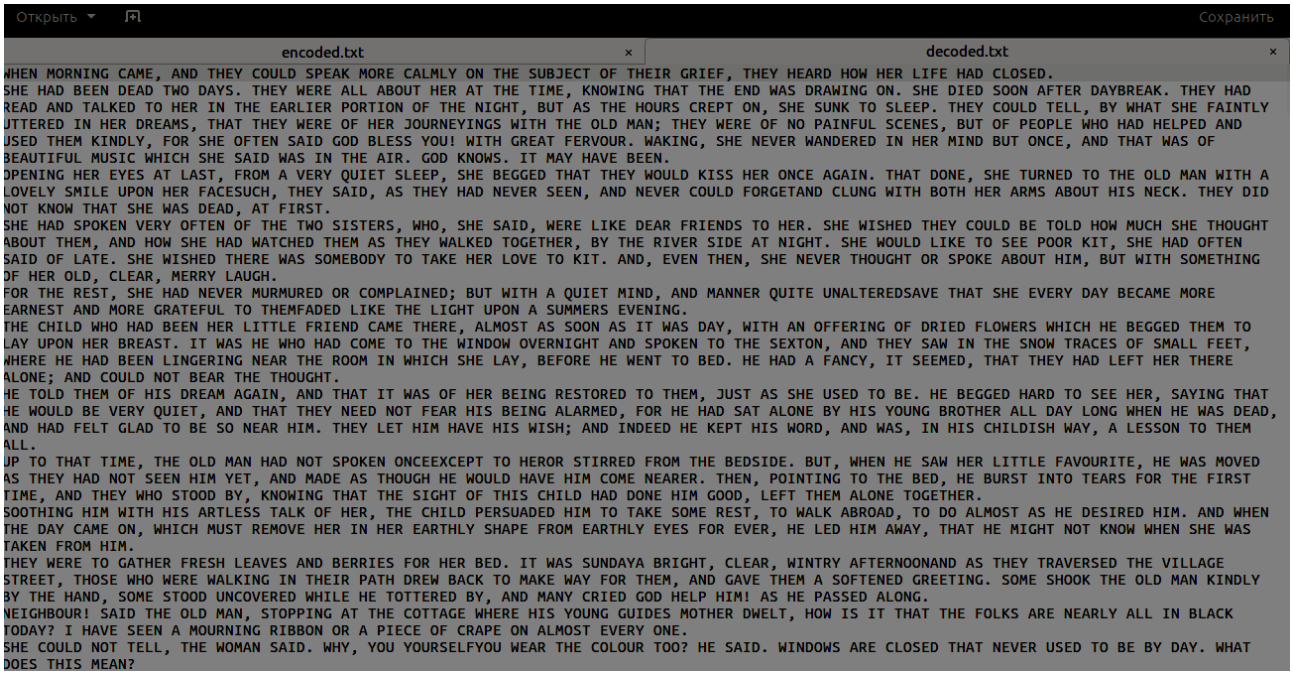

Рис.3 Сообщение после атаки на шифр Виженера

Рис.4 Ключ, полученный в результате атаки

Как видим, полученный ключ в результате атаки, совпадает с действительным ключом.