Донецкий Национальный Технический Университет

Лабораторная работа № 3

«АлгоритмЫ ШИФРОВАНИЯ»

Выполнил:

ст. группы ИПЗ -13

Лысенко А. С.

Проверила:

Скрипник Т.В.

Красноармейск 2015

Введение

Метод шифровки/дешифровки называют **шифром** (**cipher**). Некоторые алгоритмы шифрования основаны на том, что сам метод шифрования (алгоритм) является секретным. Ныне такие методы представляют лишь исторический интерес и не имеют практического значения. Все современные алгоритмы используют **ключ** для управления шифровкой и дешифровкой; сообщение может быть успешно дешифровано только если известен ключ. Ключ, используемый для дешифровки может не совпадать с ключом, используемым для шифрования, однако в большинстве алгоритмов ключи совпадают.

Алгоритмы с использованием ключа делятся на два класса: симметричные (или алгоритмы секретным ключом) и асимметричные (или алгоритмы с открытым ключом). Разница в том, что симметричные алгоритмы используют один и тот же ключ для шифрования и для дешифрования (или же ключ для дешифровки просто вычисляется по ключу шифровки). В то время как асимметричные алгоритмы используют разные ключи, и ключ для дешифровки не может быть вычислен по ключу шифровки.

Симметричные алгоритмы подразделяют на **потоковые шифры** и **блочные шифры**. Потоковые позволяют шифровать информацию побитово, в то время как блочные работают с некоторым набором бит данных (обычно размер блока составляет 64 бита) и шифруют этот набор как единое целое.

Ассиметричные шифры (также именуемые алгоритмами с открытым ключом, или --- в более общем плане --- криптографией с открытым ключом) допускают, чтобы открытый ключ был доступен всем (скажем, опубликован в газете). Это позволяет любому зашифровать сообщение. Однако расшифровать это сообщение сможет только нужный человек (тот, кто владеет ключом дешифровки). Ключ для шифрования называют **открытым ключом**, а ключ для дешифрования --- **закрытым ключом** или **секретным ключом**.

Современные алгоритмы шифровки/дешифровки достаточно сложны и их невозможно проводить вручную. Настоящие криптографические алгоритмы разработаны для использования компьютерами или специальными аппаратными устройствами. В большинстве приложений криптография производится программным обеспечением и имеется множество доступных криптографических пакетов.

Вообще говоря, симметричные алгоритмы работают быстрее, чем ассиметричные. На практике оба типа алгоритмов часто используются вместе: алгоритм с открытым ключом используется для того, чтобы передать случайным образом сгенерированный секретный ключ, который затем используется для дешифровки сообщения.

Многие качественные криптографические алгоритмы доступны широко - в книжном магазине, библиотеке, патентном бюро или в Интернет. К широко известным симметричным алгоритмам относятся DES и IDEA, Наверное самым лучшим асимметричным алгоритмом является RSA.

Основной раздел

Схема получения зашифрованного сообщения, происходит с помощью симметричного алгоритма шифрования, с использованием блочного шифра.

Сообщение шифруется и расшифровывается с помощью открытого ключа.

Ключ работает таким образом: вводим цифру от 0 до 6, показывая этим на сколько будем сдвигать вперед текущую букву в алфавите.



Программная реализация

**package** Main;  
  
**import** java.io.BufferedReader;  
**import** java.io.IOException;  
**import** java.io.InputStreamReader;  
**import** java.util.Random;  
  
**public class** Source  
{  
 **public static void** main(String[] args) **throws** IOException  
 {  
 **new** Source();  
 }  
 **private** Source()**throws** IOException  
 {  
 **this**.changeText();  
 }  
 **private void** changeText() **throws** IOException {  
 **int** i;  
 **int** index = 0;  
  
 String text;  
 Random r = **new** Random();  
 BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(System.***in***));  
 show(**"Введите текст: "**);  
 text = reader.readLine();  
 String[] mass = **new** String[text.length()];  
 **int**[] mas1 = **new int**[text.length()];  
 **char** mas55[] = **new char**[text.length()];  
 **int** nmas[] = **new int**[text.length()];  
 showLn(**"Открытый текст: "** + text);  
 showLn(**"Длина текста: "** + text.length());  
 **char**[] mas = text.toCharArray();  
 **char**[] mas4 = {  
 **'а'**, **'б'**, **'в'**, **'г'**, **'д'**, **'е'**, **'ё'**,  
 **'ж'**, **'з'**, **'и'**, **'й'**, **'к'**, **'л'**, **'м'**,  
 **'н'**, **'о'**, **'п'**, **'р'**, **'с'**, **'т'**, **'у'**,  
 **'ф'**, **'х'**, **'ц'**, **'ч'**, **'ш'**, **'щ'**,  
 **'ъ'**, **'ы'**, **'ь'**, **'э'**, **'ю'**, **'я'**, **' '**,  
 **'а'**, **'б'**, **'в'**, **'г'**, **'д'**, **'е'**, **'ё'**,  
 **'0'**, **'1'**, **'2'**, **'3'**, **'4'**, **'5'**, **'6'**, **'7'**, **'8'**, **'9'** };  
 *//while (index < mas.length)* **for** (i = 0; i < mas.**length**; i++) {  
 **if** (mas[i] == **'a'**) {  
 nmas[i] = 0;  
 index++;  
 }  
 **if** (mas[i] == **'б'**) {  
 nmas[i] = 1;  
 index++;  
 }  
 **if** (mas[i] == **'в'**) {  
 nmas[i] = 2;  
 index++;  
 }  
 **if** (mas[i] == **'г'**) {  
 nmas[i] = 3;  
 index++;  
 }  
 **if** (mas[i] == **'д'**) {  
 nmas[i] = 4;  
 index++;  
 }  
 **if** (mas[i] == **'е'**) {  
 nmas[i] = 5;  
 index++;  
 }  
 **if** (mas[i] == **'ё'**) {  
 nmas[i] = 6;  
 index++;  
 }  
 **if** (mas[i] == **'ж'**) {  
 nmas[i] = 7;  
 index++;  
 }  
 **if** (mas[i] == **'з'**) {  
 nmas[i] = 8;  
 index++;  
 }  
 **if** (mas[i] == **'и'**) {  
 nmas[i] = 9;  
 index++;  
 }  
 **if** (mas[i] == **'й'**) {  
 nmas[i] = 10;  
 index++;  
 }  
 **if** (mas[i] == **'к'**) {  
 nmas[i] = 11;  
 index++;  
 }  
 **if** (mas[i] == **'л'**) {  
 nmas[i] = 12;  
 index++;  
 }  
 **if** (mas[i] == **'м'**) {  
 nmas[i] = 13;  
 index++;  
 }  
 **if** (mas[i] == **'н'**) {  
 nmas[i] = 14;  
 index++;  
 }  
 **if** (mas[i] == **'о'**) {  
 nmas[i] = 15;  
 index++;  
 }  
 **if** (mas[i] == **'п'**) {  
 nmas[i] = 16;  
 index++;  
 }  
 **if** (mas[i] == **'р'**) {  
 nmas[i] = 17;  
 index++;  
 }  
 **if** (mas[i] == **'с'**) {  
 nmas[i] = 18;  
 index++;  
 }  
 **if** (mas[i] == **'т'**) {  
 nmas[i] = 19;  
 index++;  
 }  
 **if** (mas[i] == **'у'**) {  
 nmas[i] = 20;  
 index++;  
 }  
 **if** (mas[i] == **'ф'**) {  
 nmas[i] = 21;  
 index++;  
 }  
 **if** (mas[i] == **'х'**) {  
 nmas[i] = 22;  
 index++;  
 }  
 **if** (mas[i] == **'ц'**) {  
 nmas[i] = 23;  
 index++;  
 }  
 **if** (mas[i] == **'ч'**) {  
 nmas[i] = 24;  
 index++;  
 }  
 **if** (mas[i] == **'ш'**) {  
 nmas[i] = 25;  
 index++;  
 }  
 **if** (mas[i] == **'щ'**) {  
 nmas[i] = 26;  
 index++;  
 }  
 **if** (mas[i] == **'ъ'**) {  
 nmas[i] = 27;  
 index++;  
 }  
 **if** (mas[i] == **'ы'**) {  
 nmas[i] = 28;  
 index++;  
 }  
 **if** (mas[i] == **'ь'**) {  
 nmas[i] = 29;  
 index++;  
 }  
 **if** (mas[i] == **'э'**) {  
 nmas[i] = 30;  
 index++;  
 }  
 **if** (mas[i] == **'ю'**) {  
 nmas[i] = 31;  
 index++;  
 }  
 **if** (mas[i] == **'я'**) {  
 nmas[i] = 32;  
 index++;  
 }  
 **if** (mas[i] == **' '**) {  
 nmas[i] = 33;  
 index++;  
 }  
 **if** (mas[i] == **'0'**) {  
 nmas[i] = 41;  
 index++;  
 }  
 **if** (mas[i] == **'1'**) {  
 nmas[i] = 42;  
 index++;  
 }  
 **if** (mas[i] == **'2'**) {  
 nmas[i] = 43;  
 index++;  
 }  
 **if** (mas[i] == **'3'**) {  
 nmas[i] = 44;  
 index++;  
 }  
 **if** (mas[i] == **'4'**) {  
 nmas[i] = 45;  
 index++;  
 }  
 **if** (mas[i] == **'5'**) {  
 nmas[i] = 46;  
 index++;  
 }  
 **if** (mas[i] == **'6'**) {  
 nmas[i] = 47;  
 index++;  
 }  
 **if** (mas[i] == **'7'**) {  
 nmas[i] = 48;  
 index++;  
 }  
 **if** (mas[i] == **'8'**) {  
 nmas[i] = 49;  
 index++;  
 }  
 **if** (mas[i] == **'9'**) {  
 nmas[i] = 50;  
 index++;  
 }  
 }  
 show(**"Ключ: "**);  
 **if** (text.length() > 20) {  
 **for** (i = 0; i < text.length(); i++) {  
 mas1[i] = r.nextInt(7);  
 show(**""** + mas1[i]);  
 }  
 showLn(**" "**);  
 show(**"Шифротекст: "**);  
 **for** (i = 0; i < mas.**length**; i++) {  
 show(Character.*toString*(mas4[nmas[i] + mas1[i]]));  
 }  
 showLn(**" "**);  
 show(**"Открытый текст: "**);  
 **for** (i = 0; i < mas.**length**; i++) {  
 show(Character.*toString*(mas[i]));  
 }  
 }  
 **if** (text.length() < 20) {  
 **for** (i = 0; i < text.length(); i++) {  
 show(**"Введите ключ["** + i + **"]: "**);  
 mass[i] = reader.readLine();  
 mas1[i] = Integer.*valueOf*(mass[i]);  
 }  
 */\* for (i = 0; i < mas.length; i++) {  
 showLn(Character.toString(mas[i]) + " " + nmas[i]);  
 }\*/* showLn(**" "**);  
 **for** (i = 0; i < mas.**length**; i++) {  
 show(Character.*toString*(mas4[nmas[i] + mas1[i]]));  
 }  
 showLn(**" "**);  
 **for** (i = 0; i < mas.**length**; i++) {  
 show(Character.*toString*(mas[i]));  
 }  
 }  
 }  
 **private void** show(String txt)  
 {  
 System.***out***.print(txt);  
 }  
 **private void** showLn(String txt)  
 {  
 System.***out***.println(txt);  
 }  
  
}

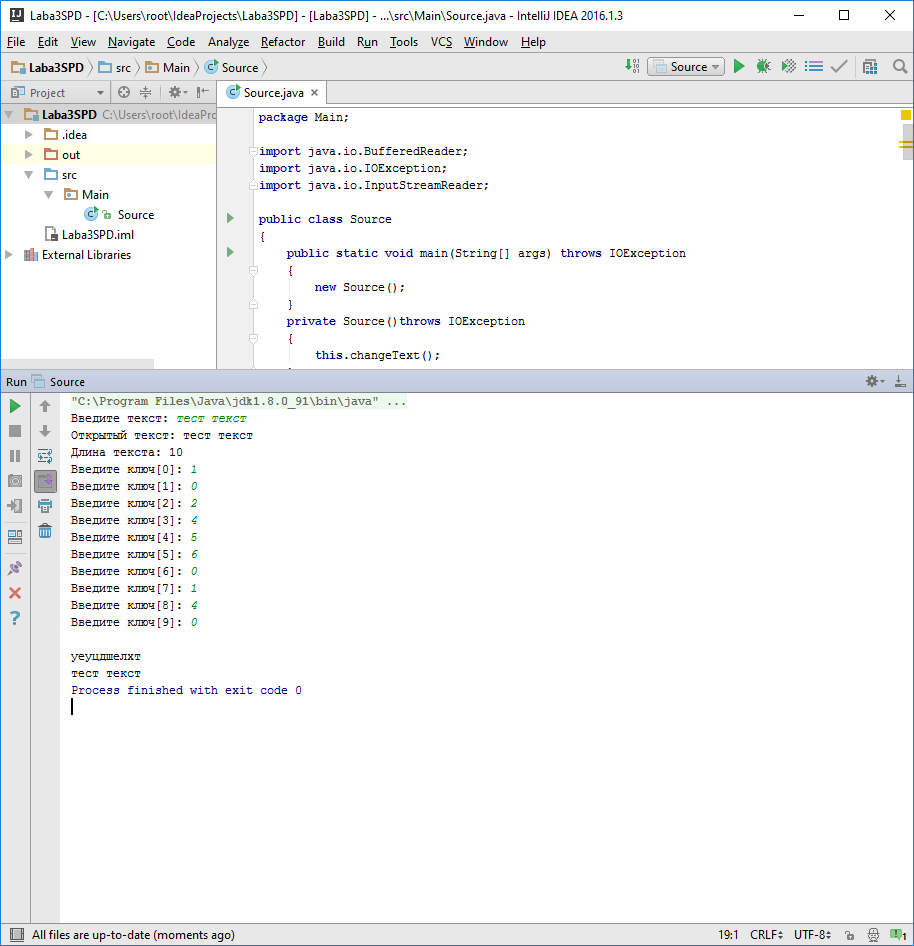


Рисунок 1

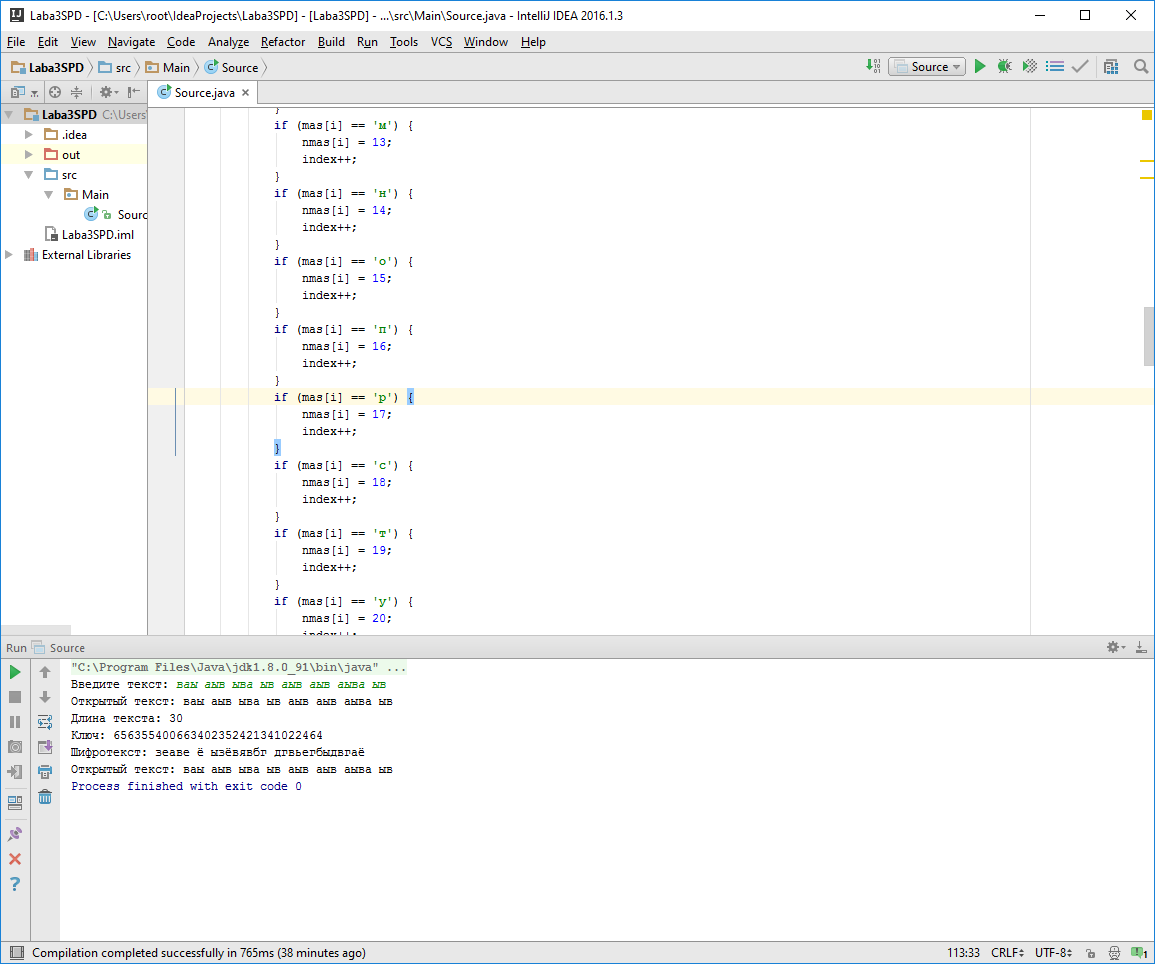


Рисунок 2

Вывод

Одноалфавитный шифр подстановки ([шифр простой замены](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80_%D0%BF%D1%80%D0%BE%D1%81%D1%82%D0%BE%D0%B9_%D0%B7%D0%B0%D0%BC%D0%B5%D0%BD%D1%8B)) — шифр, при котором каждый символ открытого текста заменяется на некоторый, фиксированный при данном ключе символ того же алфавита.