Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Рязанский государственный радиотехнический

университет имени В.Ф. Уткина»

Кафедра «ЭВМ»

Отчет о лабораторной работе №5

«Поточные шифры»

по дисциплине

«Защита информации»

Выполнили:

Студенты группы 045

Анохин В.А.

Вашкулатов Н.А.

Проверили:

доц. Крошилина С.В.

доц. Тишкина В.В.

**Цель работы**: ознакомиться с полиалфавитными шифрами на примере шифров Виженера и Вернама.

**Ход работы**

На рисунках 1 – 5 показаны схемы алгоритмов данной программы.

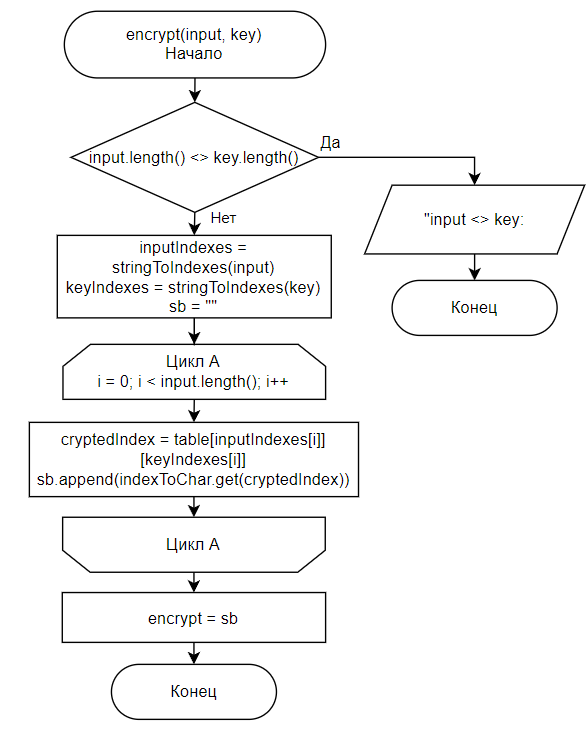


Рисунок 1 – Схема подпрограммы encrypt

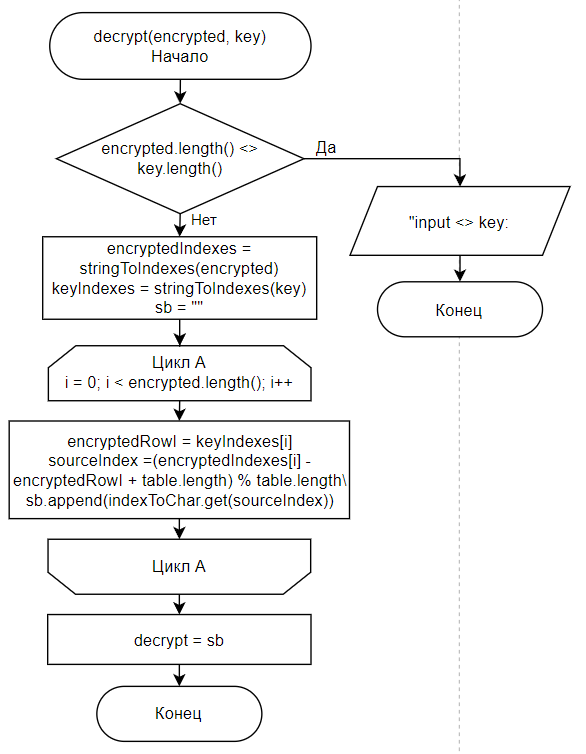


Рисунок 2 – Схема подпрограммы decrypt

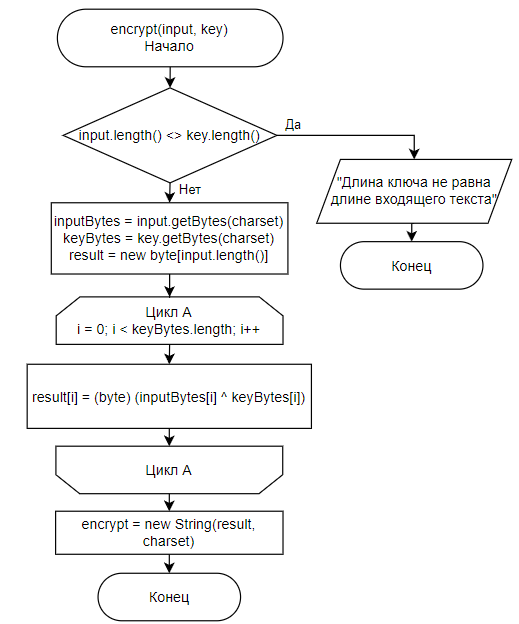


Рисунок 3 – Схема подпрограммы encrypt

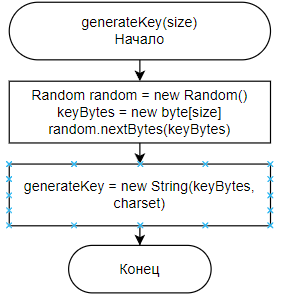


Рисунок 4 – Схема подпрограммы generateKey

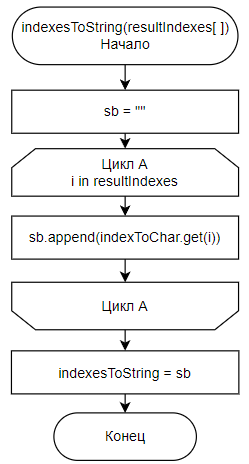


Рисунок 5 – Схема подпрограммы indexesToString

**Код программы:**

**Шифр Виженера:**

public class Vizhener {  
 private static final Map<Character, Integer> *charToIndex* = new HashMap<>();  
 private static final Map<Integer, Character> *indexToChar* = new HashMap<>();  
 static String *ALPHABET* = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя ";  
  
 private static int[][] *table*;  
  
 static {  
 int n = *ALPHABET*.length();  
 for (int i = 0; i < n; i++) {  
 *charToIndex*.put(*ALPHABET*.charAt(i), i);  
 *indexToChar*.put(i, *ALPHABET*.charAt(i));  
 }  
 *table* = new int[n][n];  
 for (int i = 0; i < n; i++) {  
 for (int j = 0; j < n; j++) {  
 *table*[i][j] = (j + i) % n;  
 }  
 }  
 }  
  
 public static String encrypt(String input, String key) {  
 if (input.length() != key.length()) throw new IllegalArgumentException("input != key");  
  
 int[] inputIndexes = *stringToIndexes*(input);  
 int[] keyIndexes = *stringToIndexes*(key);  
 StringBuilder sb = new StringBuilder();  
 for (int i = 0; i < input.length(); i++) {  
 int cryptedIndex = *table*[inputIndexes[i]][keyIndexes[i]];  
 sb.append(*indexToChar*.get(cryptedIndex));  
 }  
 return sb.toString();  
 }  
  
 public static String decrypt(String encrypted, String key) {  
 if (encrypted.length() != key.length()) throw new IllegalArgumentException("input != key");  
 int[] encryptedIndexes = *stringToIndexes*(encrypted);  
 int[] keyIndexes = *stringToIndexes*(key);  
 StringBuilder sb = new StringBuilder();  
 for (int i = 0; i < encrypted.length(); i++) {  
 int encryptedRowI = keyIndexes[i];  
 int sourceIndex = (encryptedIndexes[i] - encryptedRowI + *table*.length) % *table*.length;  
 sb.append(*indexToChar*.get(sourceIndex));  
 }  
 return sb.toString();  
 }  
  
 private static String indexesToString(List<Integer> resultIndexes) {  
 var sb = new StringBuilder();  
 for (int i : resultIndexes) {  
 sb.append(*indexToChar*.get(i));  
 }  
 return sb.toString();  
 }  
  
 private static int[] stringToIndexes(String input) {  
 int[] indexes = new int[input.length()];  
 for (int i = 0; i < input.length(); i++) {  
 if (!*charToIndex*.containsKey(input.charAt(i))) throw new IllegalArgumentException("Символ не найден " + input.charAt(i));  
 indexes[i] = *charToIndex*.get(input.charAt(i));  
 }  
 return indexes;  
 }  
}

**Шифр Вермана:**

public class Vernan {  
  
 static Charset *charset* = Charset.*forName*("Windows-1251");  
  
 public static String generateKey(int size) {  
 Random random = new Random();  
 byte[] keyBytes = new byte[size];  
 random.nextBytes(keyBytes);  
 return new String(keyBytes, *charset*);  
 }  
  
 public static String crypt(String input, String key) {  
 if (key.length() != input.length()) {  
 throw new IllegalArgumentException("Длина ключа не равна длине входящего текста");  
 }  
 byte[] inputBytes = input.getBytes(*charset*);  
 byte[] keyBytes = key.getBytes(*charset*);  
 byte[] result = new byte[input.length()];  
 for (int i = 0; i < keyBytes.length; i++) {  
 result[i] = (byte) (inputBytes[i] ^ keyBytes[i]);  
 }  
 return new String(result, *charset*);  
 }  
}

**Вывод**: ознакомиться с полиалфавитными шифрами на примере шифров Виженера и Вернама.