Міністерство освіти та науки

Прикарпатський національний університет імені Василя Стефаника

Фізико-технічний факультет

Кафедра комп’ютерної інженерії та електроніки

Лабораторна робота № 3

З курсу “Захист інформації у комп’ютерних мережах та системах”

Виконав студент групи КІ-41

Воробій Віталій

Івано-Франківськ 2020

5 варіант

|  |  |
| --- | --- |
| № варіанту | Твірна функція |
| 5 | Арифметичний зсув вправо |

Виконання роботи

Завдання виконане мовою Java і додатково використана бібліотека Junit для автоматизованого тестування коду.

Абстракція твірної функції Фейштеля:

package solution;  
  
@FunctionalInterface  
public interface FeistelFunc {  
 int calc(int R, int K);  
}

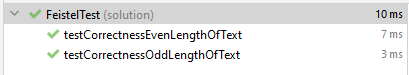
Клас, що реалізує всю логіку мережі Фейштеля.

package solution;  
  
import java.util.Random;  
  
public class Feistel {  
 private final int ROUNDS;  
 private int[] KEYS;  
 private final FeistelFunc;  
  
 public Feistel(int rounds, FeistelFunc feistelFunc) {  
 this.ROUNDS = rounds;  
 this.feistelFunc = feistelFunc;  
 initKeys();  
 }  
  
 private void initKeys() {  
 int[] keys = new int[ROUNDS];  
 Random = new Random();  
 for (int i = 0; i < ROUNDS; i++) {  
 keys[i] = random.nextInt();  
 }  
 this.KEYS = keys;  
 }  
  
 public Result cipherText(String text) {  
 boolean textIsOdd = text.length() % 2 == 1;  
 String correctText = assertTextLengthIsCorrect(text);  
 final int N = correctText.length();  
 StringBuilder builder = new StringBuilder();  
 for (int i = 0; i < N - 1; i += 2) {  
 int[] ciphered = cipher(correctText.charAt(i), correctText.charAt(i + 1));  
 builder.append((char) ciphered[0]);  
 builder.append((char) ciphered[1]);  
 }  
 return new Result(builder.toString(), textIsOdd);  
 }  
  
 public String decipherText(Result result) {  
 String text = result.getText();  
 final int N = text.length();  
 StringBuilder builder = new StringBuilder();  
 for (int i = 0; i < N - 1; i += 2) {  
 int[] ciphered = decipher(text.charAt(i), text.charAt(i + 1));  
 builder.append((char) ciphered[0]);  
 builder.append((char) ciphered[1]);  
 }  
 String textResult = builder.toString();  
 return result.getLenWasOdd() ? textResult.substring(0, N - 1) : textResult;  
 }  
  
 private String assertTextLengthIsCorrect(String text) {  
 return text.length() % 2 == 0 ? text : text + " ";  
 }  
  
 private int[] cipher(int a, int b) {  
 for (int i = 0; i < ROUNDS; i++) {  
 int t = a ^ feistelFunc.calc(b, KEYS[i]);  
 a = b;  
 b = t;  
 }  
 return new int[] {b, a};  
 }  
  
 private int[] decipher(int a, int b) {  
 for (int i = ROUNDS - 1; i >= 0; i--) {  
 int t = a ^ feistelFunc.calc(b, KEYS[i]);  
 a = b;  
 b = t;  
 }  
 return new int[] {b, a};  
 }  
  
 public static class Result {  
 private final String text;  
 private final Boolean lenWasOdd;  
  
 public Result(String text, Boolean lenWasOdd) {  
 this.text = text;  
 this.lenWasOdd = lenWasOdd;  
 }  
  
 public Boolean getLenWasOdd() {  
 return lenWasOdd;  
 }  
  
 public String getText() {  
 return text;  
 }  
  
 @Override  
 public String toString() {  
 return text;  
 }  
 }  
  
}

Тест Junit для наведеного вище коду:

package solution;  
  
import org.junit.BeforeClass;  
import org.junit.Test;  
import org.junit.jupiter.api.BeforeAll;  
  
import static org.junit.jupiter.api.Assertions.*assertEquals*;  
  
public class FeistelTest {  
 private static Feistel *feistel*;  
  
 @BeforeClass  
 public static void setUp() {  
 *feistel* = new Feistel(10, (R, K) -> R << K); *// Число раундів і твірна функція* }  
  
 @Test  
 public void testCorrectnessOddLengthOfText() {  
 String text = "Hello world";  
 *assertEquals*(text.length() % 2, 1);  
  
 Feistel.Result cipherResult = *feistel*.cipherText(text);  
  
 *assertEquals*(cipherResult.getLenWasOdd(), true);  
 *assertEquals*(*feistel*.decipherText(cipherResult), text);  
 }  
  
 @Test  
 public void testCorrectnessEvenLengthOfText() {  
 String text = "SomeText";  
 *assertEquals*(text.length() % 2, 0);  
  
 Feistel.Result cipherResult = *feistel*.cipherText(text);  
  
 *assertEquals*(cipherResult.getLenWasOdd(), false);  
 *assertEquals*(*feistel*.decipherText(cipherResult), text);  
 }  
  
}

Результат проходження тестів:



Клас для демонстрації роботи програми:

package solution;  
  
import java.util.Scanner;  
  
public class FeistelDemo {  
 private static final String *STOP\_TEXT* = "stop";  
 private static final Feistel *feistel* = new Feistel(10, (R, K) -> R << K);  
  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);

System.*out*.println("Введіть текст");  
 while (scan.hasNext()) {  
 String text = scan.nextLine();  
 if (text.toLowerCase().equals(*STOP\_TEXT*)) {  
 break;  
 }  
 Feistel.Result cipherResult = *feistel*.cipherText(text);  
 String decipheredText = *feistel*.decipherText(cipherResult);  
 System.*out*.println("Вхідний текст: " + text);  
 System.*out*.println("Зашифрований текст: " + cipherResult);  
 System.*out*.println("Розшифрований текст: " + decipheredText);  
 System.*out*.println("Розшифрування пройшло успішно: " + (text.equals(decipheredText) ? "Так" : "Ні"));  
 System.*out*.println();  
 }  
 System.*out*.println("Програму завершено");  
 }  
  
}

Демонстрація роботи програми:



Висновок: на цій лабораторній роботі я навчився реалізовувати мережі Фейштеля.