Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Отчёт**

по криптографическим методам защиты информации

**Лабораторная работа №7**

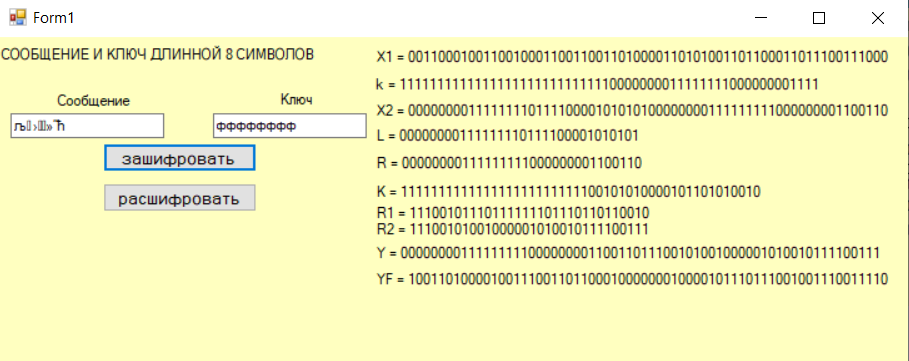
**Тема «**Исследование блочных шифров**»**

Цель: изучение и приобретение практических навыков разработки и использования приложений для реализации блочных шифров

Листинг приложения

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  namespace des  {  public partial class Form1 : Form  {  public string X, X1, k;  public int[] P = new int[]  {  16, 7, 20, 21, 29, 12, 28, 17,  1, 15, 23, 26, 5, 18, 31, 10,  2, 8, 24, 14, 32, 27, 3, 9,  19, 13, 30, 6, 22, 11, 4, 25  };  public int[] IP = new int[]  {58, 50, 42, 34, 26, 18, 10, 2, 60, 52, 44, 36, 28, 20, 12, 4,  62, 54, 46, 38, 30, 22, 14, 6, 64, 56, 48, 40, 32, 24, 16, 8,  57, 49, 41, 33, 25, 17, 9, 1, 59, 51, 43, 35, 27, 19, 11, 3,  61, 53, 45, 37, 29, 21, 13, 5, 63, 55, 47, 39, 31, 23, 15, 7 };  public int[] IP1 = new int[]  {40, 8, 48, 16, 56, 24, 64, 32, 39, 7, 47, 15, 55, 23, 63, 31,  38, 6, 46, 14, 54, 22, 62, 30, 37, 5, 45, 13, 53, 21, 61, 29,  36, 4, 44, 12, 52, 20, 60, 28, 35, 3, 43, 11, 51, 19, 59, 27,  34, 2, 42, 10, 50, 18, 58, 26, 33, 1, 41, 9, 49, 17, 57, 25 };  public int[] E = new int[] { 32, 1, 2, 3, 4, 5,  4, 5, 6, 7, 8, 9,  8, 9, 10, 11, 12, 13,  12, 13, 14, 15, 16, 17,  16, 17, 18, 19, 20, 21,  20, 21, 22, 23, 24, 25,  24, 25, 26, 27, 28, 29,  28, 29, 30, 31, 32, 1 };  static public int[] PC1 = new int[]  {57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18,  10, 2, 59, 51, 43, 35, 27, 19, 11, 3, 60, 52, 44, 36,  63, 55, 47, 39, 31, 23, 15, 7, 62, 54, 46, 38, 30, 22,  14, 6, 61, 53, 45, 37, 29, 21, 13, 5, 28, 20, 12, 4 };  static int[,] s1 = new int[,]  {  {14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7 },  { 0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8 },  { 4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0 },  { 15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13 } };  static int[,] s2 = new int[,]  {  { 15, 1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13, 12, 0, 5, 10 },  { 3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5 },  { 0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15 },  { 13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9 } };  static int[,] s3 = new int[,]  {  { 10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8 },  { 13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1 },  { 13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7 },  { 1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12 } };  static int[,] s4 = new int[,]  {  { 7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15 },  { 13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9 },  { 10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4 },  { 3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14 } };  static int[,] s5 = new int[,]  {  { 2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9 },  { 14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6 },  { 4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14 },  { 11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3 } };  static int[,] s6 = new int[,]  {  { 12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11 },  { 10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8 },  { 9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6 },  { 4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13 } };  static int[,] s7 = new int[,]  {  { 4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1 },  { 13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6 },  { 1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2 },  { 6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12 } };  static int[,] s8 = new int[,]  {  { 13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7 },  { 1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2 },  { 7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8 },  { 2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11 } };  public int[] PC2 = new int[]  { 14, 17, 11, 24, 1, 5, 3, 28,  15, 6, 21, 10, 23, 19, 12, 4,  26, 8, 16, 7, 27, 20, 13, 2,  41, 52, 31, 37, 47, 55, 30, 40,  51, 45, 33, 48, 44, 49, 39, 56,  34, 53, 46, 42, 50, 36, 29, 32 };  static public int[] shiftKey = new int[] { 1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1 };  static List<int[,]> listSMatrix = new List<int[,]> { s1, s2, s3, s4, s5, s6, s7, s8 };  public string ChangeSMatrix(string mess6Bit, int numSMatrix)  {  StringBuilder temp = new StringBuilder();  int adrX = Convert.ToInt32(temp.Append(mess6Bit[0]).Append(mess6Bit[5]).ToString(), 2);  int adrY = Convert.ToInt32(mess6Bit.Substring(1, 4), 2);  string res = Convert.ToString(listSMatrix[numSMatrix][adrX, adrY], 2);  if (res.Length < 4)  res = res.PadLeft(4, '0');  return res;  }  public Form1()  {  InitializeComponent();  }  /// <summary>  ///  /// </summary>  /// <param name="sender"></param>  /// <param name="e"></param>  private void button1\_Click(object sender, EventArgs e)  {  if (textBox1.Text.Length == 8 && textBox2.Text.Length == 8)  {  X = textBox1.Text;  X1 = BinaryConvert.ToBinary(BinaryConvert.ConvertToByteArray(X, Encoding.Default));  k = Permut(BinaryConvert.ToBinary(BinaryConvert.ConvertToByteArray(textBox2.Text, Encoding.Default)), PC1);  label0.Text = "X1 = " + X1;  label1.Text = "k = " + k;  string X2 = Permut(X1, IP);  label2.Text = "X2 = " + X2;  string L, R, L1, R1, R2, Y, K;  //round  L = X2.Substring(0, 32);  R = X2.Substring(32);  label3.Text = "L = " + L;  label4.Text = "R = " + R;  K = GenKey(k, 1);  label5.Text = "K = " + K;  R1 = FFunc(R, K);  label6.Text = "R1 = " + R1;  L1 = R;  R2 = XOR(L, R1);  label7.Text = "R2 = " + R2;  Y = String.Concat(L1, R2);  label8.Text = "Y = " + Y;  //end round  Y = Permut(Y, IP1);  label9.Text = "YF = " + Y;  textBox1.Text = BinaryConvert.ConvertBytesToString(BinaryConvert.GetBytes(Y), Encoding.Default);  }  }  //==============================================================  private void button2\_Click(object sender, EventArgs e)  {  if (textBox1.Text.Length == 8 && textBox2.Text.Length == 8)  {  X = textBox1.Text;  X1 = BinaryConvert.ToBinary(BinaryConvert.ConvertToByteArray(X, Encoding.Default));  k = Permut(BinaryConvert.ToBinary(BinaryConvert.ConvertToByteArray(textBox2.Text, Encoding.Default)), PC1);  label0.Text = "X1 = " + X1;  label1.Text = "k = " + k;  string X2 = Permut(X1, IP);  label2.Text = "X2 = " + X2;  string L, R, L1, R1, L2, Y, K;  //round  L = X2.Substring(0, 32);  R = X2.Substring(32);  label3.Text = "L = " + L;  label4.Text = "R = " + R;  K = GenKey(k, 1);  label5.Text = "K = " + K;  L1 = FFunc(L, K);  label6.Text = "L1 = " + L1;  R1 = L;  L2 = XOR(R, L1);  label7.Text = "L2 = " + L2;  Y = String.Concat(L2, R1);  label8.Text = "Y = " + Y;  //end round  Y = Permut(Y, IP1);  label9.Text = "YF = " + Y;  textBox1.Text = BinaryConvert.ConvertBytesToString(BinaryConvert.GetBytes(Y), Encoding.Default);  }  }  public string GenKey(string oldKey56, int round)  {  var temp1 = oldKey56.Substring(0, 28);  var temp2 = oldKey56.Substring(28);  StringBuilder key1 = new StringBuilder();  key1.Append(temp1.Substring(temp1.Length - shiftKey[round - 1]));  key1.Append(temp1.Substring(shiftKey[round - 1]));  StringBuilder key2 = new StringBuilder();  key2.Append(temp2.Substring(temp2.Length - shiftKey[round - 1]));  key2.Append(temp2.Substring(shiftKey[round - 1]));  StringBuilder sb = new StringBuilder();  foreach (var item in PC2)  {  sb.Append(key1.Append(key2)[item - 1]);  }  return sb.ToString();  }  public string Permut(string str, int[] matrix)  {  StringBuilder sb = new StringBuilder();  foreach (var item in matrix)  {  sb.Append(str[item - 1]);  }  return sb.ToString();  }  private void label10\_Click(object sender, EventArgs e)  {  }  private void label13\_Click(object sender, EventArgs e)  {  }  public string FFunc(string mess32Bit, string key)  {  StringBuilder resStr = new StringBuilder();  string mess = XOR(Permut(mess32Bit, E), key);  for (int i = 0; i < mess.Length; i = i + 6)  {  resStr.Append(ChangeSMatrix(mess.Substring(i, 6), i / 6));  }  return Permut(resStr.ToString(), P);  }  public string XOR(string value1, string value2)  {  StringBuilder sb = new StringBuilder();  for (int i = 0; i < value1.Length; i++)  {  if (value1[i].Equals(value2[i]))  sb.Append("0");  else  sb.Append("1");  }  return sb.ToString();  }    }  } |

Пример работы приложения



**Вывод**

В ходе лабораторной работы были усвоены теоретические знания по алгоритмам блочного шифрования, а так же разработано приложение для расшифрования и зашифрования сообщений по алгоритму DES.